MICRO-INFORMATIQUE



QUICK REFERENCE

MS-DOS FICHIERS BATCH

MICROSOFT PRESS

Traitement par lots • Fichiers de commandes • Paramètres • ANSI.SYS et DEBUG • Concepts avancés • Bibliothèque de fichiers • Optimisation d'AUTOEXEC.BAT

PRESSES POCKET

QUICK REFERENCE
MS-DOS
FICHIERS BATCH

AVERTISSEMENT

Traduction de l'ouvrage américain :
MS-DOS Batch Files
publié par Microsoft Press, une division de Microsoft Corporation
16011 NE 36th Way, Box 97017, Redmond, Washington 98073-9717

ISBN : 2-266-04059-6
ISSN : 1150-305X

MICRO-INFORMATIQUE

KRIS JAMSA

Texte français et mise en page
Jean Guillemin

QUICK REFERENCE MS-DOS FICHIERS BATCH



A VOTRE SERVICE

CATALOGUES ET "LIVRES MICRO"

Je désire recevoir gratuitement : ❐ votre catalogue général
❐ la revue "Livres Micro" (avec abonnement gratuit)

VOTRE AVIS NOUS INTERESSE

Pour nous permettre de faire de meilleurs livres, adressez-nous vos critiques et suggestions sur le présent ouvrage.

Titre de l'ouvrage : **MS-DOS fichiers BATCH**

Ce livre vous donne-t-il toute satisfaction ? ______________________

Avez-vous des commentaires à formuler ? ______________________

____

____

Avez-vous déjà acquis des livres P.S.I ? __________ Si oui, lesquels ? ______

____

Qu'en pensez-vous ? ______________________

Où les avez-vous achetés ? ❐ Librairie ❐ Boutique micro
❐ Par correspondance

Votre centre d'intérêt ? ❐ PC (ou compatibles) ❐ Macintosh
❐ Atari ❐ Autre __________

Nom ______________________

Prénom ______________________ Age __________

Adresse ______________________

Code Postal __________ Ville __________

Profession ______________________

A DECOUPER ET A ENVOYER AUX : **Editions P.S.I., Courrier Lecteur**
3 avenue Gallieni
94257 GENTILLY Cedex

Sommaire

Introduction

Depuis des années, les utilisateurs expérimentés du système d'exploitation MS-DOS apprécient les économies de temps et de frappe qu'ils doivent aux fichiers de traitement par lots (*batch files*). Un fichier DOS de traitement par lots, ou fichier de commandes, est tout simplement un fichier qui contient des commandes DOS. Quand vous tapez le nom d'un tel fichier après l'indicatif MS-DOS (*prompt*), le système d'exploitation exécute les commandes contenues dans ce fichier (ce livre utilise C> comme indicatif par défaut). De nombreux utilisateurs emploient des fichiers de commandes simples de façon quotidienne, pour abréger une série de commandes courantes. Bien peu cependant tirent parti du potentiel considérable offert par les fichiers de commandes.

Jusqu'à présent, la plupart des ouvrages sur le «traitement par lots» étaient largement consacrés aux commandes DOS, en dehors du traitement par lots. En revanche, ce livre ne quitte pas son domaine. Les utilisateurs débutants des fichiers de commandes peuvent commencer au chapitre 1. Ceux qui connaissent les fichiers de commandes simples apprendront à répéter des commandes et à effectuer des branchements d'un point à un autre d'un fichier de commandes. Les utilisateurs expérimentés apprendront à ajouter la couleur et les caractères de tracé de lignes aux écrans produits par leurs fichiers de commandes.

Les fichiers de commandes DOS sont très puissants. Pourtant, les commandes ne vous permettent pas à elles seules de tirer tout le bénéfice de vos fichiers de traitement par lots. Ce guide de référence rapide vous montre comment créer, à l'aide de DEBUG, des programmes simples qui inciteront l'utilisateur à fournir une réponse O (oui) ou N (non), qui

détecteront la pression d'une certaine touche de fonction et même utiliseront les touches de direction pour changer d'option dans un menu. L'élaboration de ces programmes ne vous demandera que quelques minutes. Aucun langage de programmation n'est exigé. Vous n'avez besoin de rien, en dehors de DOS !

Ce livre vous apprendra aussi comment procéder pour que votre fichier de commandes sollicite des informations de l'utilisateur et sache ensuite lire et exploiter les informations fournies. Depuis des années, les utilisateurs ont souhaité disposer de ces fonctions. Ce guide de référence rapide vous montre comment les mettre en place.

Vous n'avez pas besoin d'être un expert pour vous servir de ce guide. Toute personne qui emploie DOS pourra suivre les exemples sans difficulté. Les moments que vous passerez devant votre écran pour apprendre à créer et exploiter les fichiers de commandes pourraient bien se révéler les plus productifs, car grâce à eux vous gagnerez du temps à chaque utilisation de l'ordinateur.

Note : ce livre a pour base la version 4.0 de DOS ; les exceptions sont signalées.

CHAPITRE 1

Principes du traitement par lots

FONCTIONNEMENT DU TRAITEMENT PAR LOTS

Le nombre de commandes DOS est maintenant supérieur à 60. La plupart d'entre nous, cependant, n'utilisent qu'une poignée de commandes DOS, telles que COPY, REN, DEL, TYPE, DISKCOPY, FORMAT et éventuellement les commandes relatives aux sous-répertoires. Il en résulte que nous oublierons probablement la syntaxe de beaucoup d'autres commandes, sans parler des commutateurs affectant leur ligne de commande.

Pour accomplir des tâches spécifiques comme le choix d'un certain répertoire et le lancement d'un traitement de texte, l'utilisateur novice est souvent amené à exécuter une série de commandes. Plus le volume de frappe demandé à l'utilisateur s'accroît et plus la probabilité d'erreur augmente, l'erreur pouvant aller de la simple faute de frappe à l'omission d'une commande critique.

Pour diminuer le risque d'erreur et simplifier votre travail, DOS vous offre le fichier *batch* ou fichier de traitement par lots, également appelé fichier de commandes. Un fichier de commandes est un fichier que vous créez et qui contient une ou plusieurs commandes DOS. Pour exécuter toutes les commandes d'un fichier de traitement par lots, il suffit de taper le nom de ce fichier après l'indicatif DOS et d'appuyer sur Entrée, exactement comme si le fichier de commandes était

composé d'une seule commande DOS. Quand DOS rencontre le fichier de commandes, il exécute toutes les commandes de la séquence, de la première à la dernière.

Pour commencer, examinons un fichier de commandes simple, appelé TIMEDATE.BAT, qui contient trois commandes élémentaires de DOS :

```
CLS
TIME
DATE
```

Vous pourrez créer ce fichier ultérieurement. Pour l'instant, voyons seulement comment il fonctionne.

CLS efface l'écran. TIME affiche l'heure courante et invite l'utilisateur à changer l'heure si nécessaire. DATE affiche la date du jour et invite l'utilisateur à la modifier s'il y a lieu.

Remarquez l'extension BAT au nom de fichier. L'extension à un nom de fichier permet de décrire le type d'information contenue dans le fichier. Dans le cas de TIMEDATE.BAT, l'extension BAT signale à DOS comme à l'utilisateur que le fichier est un fichier de traitement par lots (BAT est l'abréviation de *batch* ou lot). Pour exécuter le fichier de commandes, il suffirait de taper son nom après l'indicatif DOS et d'appuyer ensuite sur Entrée, comme suit :

```
C> TIMEDATE
```

Quand DOS rencontre le nom du fichier de traitement par lots, il ouvre le fichier et exécute la première commande (dans ce cas, la commande CLS). Lorsque l'exécution de CLS est terminée, DOS lance la commande suivante, c'est-à-dire TIME. A cet instant, DOS affiche :

```
C> TIME
L'heure courante est : 10:23:17,45
Donnez la nouvelle heure :
```

Il faut maintenant taper l'heure réelle et appuyer sur la touche Entrée ; ou simplement appuyer sur Entrée pour laisser l'heure inchangée (note : le format de la commande

TIME peut être modifié par la commande COUNTRY dans le fichier CONFIG.SYS). Quand l'exécution de TIME est achevée, DOS exécute la commande DATE et affiche :

```
C> DATE
La date courante est : Lun 29/01/90
Donnez la nouvelle date (jj/mm/aa) :
```

Vous pouvez à présent taper la date correcte et appuyer sur Entrée ; ou conserver la date affichée en appuyant seulement sur Entrée. Quand la commande DATE a terminé son exécution, DOS cherche la commande suivante dans le fichier de traitement par lots. Comme il n'existe plus aucune commande, DOS met un terme à l'exécution du fichier et affiche son indicatif.

Examinons un autre fichier de traitement par lots appelé INFODISK, contenant trois commandes DOS :

```
VOL
CHKDSK
DIR
```

Pour exécuter les commandes de ce fichier, qui affiche des informations spécifiques du lecteur par défaut, il suffirait de frapper le nom du fichier et d'appuyer sur Entrée.:

```
C> INFODISK
```

Si DOS affiche le message *Nom de commande ou de fichier incorrect* quand vous exécutez le fichier de commandes INFODISK.BAT, cela signifie que DOS n'est pas parvenu à localiser la commande externe CHKDSK.COM. Rappelez-vous : les commandes externes doivent résider sur la disquette, dans les lecteur et répertoire courants ou être mentionnées dans le chemin d'accès défini par la commande PATH de DOS.

Note pour les utilisateurs d'OS/2

Les fichiers de commandes ne sont pas réservés uniquement à DOS. Les fichiers de commandes d'OS/2 ont le même

comportement que ceux de DOS. La seule différence entre les fichiers de commandes de DOS et d'OS/2 est la dénomination. Le mode réel d'OS/2 vous permet d'exécuter des commandes DOS ainsi que des fichiers de commandes DOS avec l'extension BAT. Dans le mode protégé d'OS/2, en revanche, les fichiers de commandes ont l'extension CMD. Dans ce cas, CMD est une abréviation de «command». Dans l'hypothèse où vous utilisez le mode protégé d'OS/2, vous pouvez créer un fichier appelé TIMEDATE.CMD qui contient trois commandes DOS (CLS, TIME et DATE).

Comme précédemment, pour exécuter le fichier de commandes, tapez le nom du fichier après l'indicatif d'OS/2 et appuyez sur la touche Entrée :

```
[c:\] TIMEDATE
```

Quand OS/2 rencontre le fichier, il exécute les commandes que celui-ci contient, de la première à la dernière.

AVANTAGES DES FICHIERS DE COMMANDES

Que vous utilisiez DOS ou OS/2, les fichiers de commandes vous font gagner du temps, réduisent le volume de frappe et le nombre d'erreurs, tout en simplifiant l'exécution de commandes difficiles. Regardons quelques exemples de fichiers de commandes illustrant ces avantages.

Gagnez du temps grâce aux fichiers de commandes

Supposons que chaque matin vous soyez amené à exécuter quatre programmes de contrôle de stock. Le premier programme, CALCSTOC.EXE, calcule la valeur actuelle de votre stock de produits. Le deuxième, TRISTOC.EXE, trie le stock par quantité disponible. Le troisième programme, IMPSTOC.EXE, fournit des listes de quantités disponibles.

Le quatrième, ACHASTOC.EXE, élabore les commandes d'approvisionnement des articles à stocker.

Pour faire tourner ces quatre programmes, il faudrait en principe taper le nom du premier, appuyer sur la touche Entrée et attendre que l'exécution de ce programme soit terminée, avant de répéter ce processus pour les trois autres programmes. Vous seriez amené à passer un temps considérable à attendre devant votre clavier que chaque programme ait achevé son exécution.

Ces commandes sont d'excellentes candidates à la constitution d'un fichier de commandes qui sera appelé, par exemple, TOUSTOC.BAT. Ce fichier contient les quatre commandes suivantes : CALCSTOC, TRISTOC, IMPSTOC et ACHASTOC. Quand vous tapez le nom du fichier de commandes et appuyez sur Entrée, DOS se charge d'exécuter pour vous successivement chacune des commandes :

```
C> TOUSTOC

C> CALCSTOC

C> TRISTOC

C> IMPSTOC

C> ACHASTOC
```

Du fait que DOS lance les commandes automatiquement à votre place, vous êtes disponible pour accomplir d'autres tâches sans vous occuper de votre ordinateur. C'est ainsi qu'un fichier de commandes peut vous faire économiser chaque jour un temps considérable.

Réduisez la frappe et les erreurs grâce aux fichiers de commandes

Puisque les fichiers de commandes vous permettent d'exécuter des commandes multiples en n'entrant qu'un seul nom de commande, ils réduisent le nombre de touches que l'utilisateur aura à frapper, ce qui diminue directement la probabilité

d'erreur. Dans l'exemple précédent, l'exécution du fichier ACHASTOC.BAT non seulement réduisait le nombre de frappes, mais encore éliminait la possibilité pour l'utilisateur de commettre une erreur dans l'ordre des commandes, de taper une commande incorrecte ou d'oublier une commande.

Simplifiez l'exécution des commandes

Chacun de nous a été, un jour, débutant de DOS. Pour la plupart, nous sommes capables de nous rappeler la timidité ressentie au moment d'émettre nos premières commandes DOS. Les fichiers de commandes atténuent ce «trac» en réduisant le nombre de commandes difficiles qu'un novice doit mémoriser et exécuter avec succès. Par exemple, la plupart des utilisateurs gardent leurs fichiers de traitement de texte dans un sous-répertoire unique. Pour lancer le traitement de texte, il faut tout d'abord choisir le sous-répertoire correct à l'aide de la commande CHDIR (*Change Directory*) et ensuite exécuter la commande du traitement de texte. Dans le cas de Microsoft Word, par exemple, la séquence des commandes devient :

```
C> CHDIR \WORD
C> WORD
```

Que vous soyez débutant ou utilisateur expérimenté aidant un novice, vous envisagerez peut-être de créer un fichier de commandes appelé DEMWORD.BAT, contenant les deux commandes. Moins le débutant aura de commandes à mémoriser, plus il se sentira à l'aise devant l'ordinateur et plus son apprentissage sera rapide.

NOMS DES FICHIERS DE COMMANDES

Il est bon de chercher à donner le maximum de sens au nom que vous donnez à chaque fichier créé sur le disque. Les fichiers de commandes ne font pas exception à cette règle.

Les fichiers de commandes doivent avoir l'extension BAT sous DOS et l'extension CMD sous OS/2. Par suite, vous n'avez que les huit caractères des noms de fichiers pour distinguer un fichier de commandes d'un autre. Le nom du fichier de commandes devrait expliquer clairement le traitement effectué par ce fichier. Par exemple, nous avons examiné auparavant le fichier de commandes TIMEDATE.BAT, qui permettait de régler l'heure et la date de votre système ; dans notre exemple de suivi de stock, le fichier de commandes avait pour nom TOUSTOC.BAT. Ces deux noms de fichiers de commandes expliquent le traitement effectué par le fichier. Appeler un fichier de commandes X.BAT ou Y.BAT est certes facile, mais aucun de ces noms n'explique le rôle du fichier. Quelques jours après sa création, vous avez déjà oublié sa fonction.

Ne donnez jamais à un fichier de commandes le nom d'une commande DOS interne ou externe. Chaque fois que vous tapez une commande après l'indicatif DOS, DOS vérifie d'abord s'il s'agit d'une commande interne (comme CLS, DATE ou TIME) que DOS retient en mémoire en permanence. Si votre commande est une commande interne, DOS l'exécute (au lieu d'exécuter le fichier de commandes). Dans le cas contraire, DOS fait un test de reconnaissance d'une commande externe dans le répertoire courant. Une commande DOS externe (telle que DISKCOPY ou FORMAT) est une commande qui réside sur disque. Si votre commande n'est ni une commande interne (en mémoire) ni une commande externe (sur disque), DOS teste la coïncidence entre votre commande et un fichier de commandes DOS. Le système (DOS ou OS/2) n'exécutera un fichier de commandes que si son nom ne correspond ni à une commande interne, ni à une commande externe des lecteur et répertoire courants. Si vous donnez à votre fichier de commandes un nom identique à celui d'une commande DOS, DOS n'exécutera jamais le fichier de commandes. Par exemple, si votre fichier de commandes s'appelle TIME.BAT, DOS localise la

commande TIME et n'exécute que la commande DOS correspondante.

CRÉATION DES FICHIERS DE COMMANDES

La méthode à choisir pour créer des fichiers de commandes peut varier selon leur longueur. Pour de courts fichiers de commandes, la méthode la plus simple et la plus rapide consiste à copier le fichier de commandes à partir du clavier. Pour cela, vous effectuez une opération de copie qui utilise CON (le nom de périphérique attribué à la *console*, c'est-à-dire au clavier) comme source des données d'entrée dans votre fichier de commandes. Dans ce cas, nous créerons le fichier de commandes TIMEDATE.BAT qui règle l'heure et la date du système. Pour commencer l'opération de copie du fichier de commandes, tapez la commande suivante après l'indicatif DOS :

```
C> COPY CON TIMEDATE.BAT
```

DOS effectuera une opération de copie fondée sur la commande COPY. Dans ce cas, la source des données à copier est le clavier. La cible de l'opération de copie est le fichier de commandes TIMEDATE.BAT. Quand vous appuyez sur la touche Entrée pour commencer la copie, DOS place le curseur au début de la ligne qui suit la ligne de commande COPY. A cet instant, DOS attend la première ligne de données. Tapez la commande TIME, qui est la première ligne du fichier de commandes, et appuyez sur Entrée :

```
C> COPY CON TIMEDATE.BAT
TIME
```

Tapez ensuite la commande DATE et appuyez sur Entrée :

```
C> COPY CON TIMEDATE.BAT
TIME
DATE
```

La commande DATE est la dernière commande du fichier. Vous devez signifier à DOS que vous avez épuisé toutes les entrées destinées au fichier. Pour ce faire, appuyez sur la touche de fonction F6 (fin de fichier) puis sur Entrée. DOS affichera les caractères ^Z à la fin de votre fichier de commandes (attention : vous pouvez, au lieu d'utiliser F6, maintenir la touche Ctrl enfoncée et taper Z). Les caractères ^Z, qui s'énoncent «Contrôle Z», indiquent à DOS la fin du fichier. Quand vous appuyez sur Entrée après ^Z, vous informez DOS que l'opération de copie est terminée. Il en résulte que DOS crée le fichier de commandes et affiche :

```
C> COPY CON TIMEDATE.BAT
TIME
DATE
^Z
    1 Fichier(s) copié(s)
C>
```

Pour exécuter ce fichier de commandes, tapez TIMEDATE après l'indicatif DOS :

```
C> TIMEDATE
```

Quand vous appuyez sur Entrée, DOS exécute la commande TIME, puis la commande DATE.

En utilisant la technique COPY CON, créez un fichier de commandes appelé INFODISK.BAT et contenant les commandes VOL, CHKDSK et DIR. Comme précédemment, tapez la ligne de commande COPY en utilisant le nom de périphérique CON en tant que source de vos données d'entrée. Appuyez ensuite sur Entrée :

```
C> COPY CON INFODISK.BAT
```

Tapez ensuite chacune des commandes composant le fichier, en appuyant sur Entrée après chaque nom de commande :

```
C> COPY CON INFODISK.BAT
VOL
CHKDSK
DIR
```

Pour indiquer à DOS que vous n'avez plus de commandes à entrer, appuyez sur la touche F6, puis sur la touche Entrée. DOS crée le fichier de commandes et vous signale qu'un fichier a été copié :

```
C> COPY CON INFODISK.BAT
VOL
CHKDSK
DIR
^Z
        1 Fichier(s) copié(s)
C>
```

Pour créer de petits fichiers de commandes à partir du clavier, suivez les étapes ci-dessous :

1. Tapez une commande COPY utilisant comme source des données le clavier (CON) et comme cible un fichier de commandes dont le nom évoque la fonction. Appuyez ensuite sur Entrée. Par exemple :

   ```
   C> COPY CON IMPRIDAT.BAT
   ```

2. Tapez les commandes du fichier une par une et appuyez sur Entrée après chacune d'elles.

3. Après avoir tapé la dernière des commandes du fichier, appuyez sur la touche F6, puis sur Entrée (F6 indique à DOS la fin du fichier).

Si vos fichiers de commandes prennent de l'ampleur, vous préférerez utiliser soit un traitement de texte, soit l'éditeur ligne EDLIN fourni par DOS. Si vous employez un traitement de texte, assurez-vous que le fichier est sauvegardé en mode non structuré ou texte seul (ASCII). Vous l'avez sans doute remarqué, les traitements de texte vous permettent de mettre en forme une lettre ou un rapport en alignant les paragraphes, en centrant le texte, ou en attirant l'attention sur des portions spécifiques du texte. Pour obtenir ces résultats, les programmes de traitement de texte incorporent à votre fichier des caractères de mise en forme. Ces caractères

ont une signification pour le programme de traitement de texte, mais ils ne sont pas compris par DOS. Si on les voyait apparaître dans vos fichiers de commandes, ils produiraient des erreurs. Si vous ouvrez ou sauvegardez vos fichiers de traitement de texte en mode ASCII, le programme n'introduira aucun caractère de mise en forme.

Lorsque vous affichez la liste des fichiers de votre répertoire DOS (dans les systèmes à disque dur) ou de votre disquette DOS (sur les systèmes à disques souples), vous observez que DOS contient un fichier sur disque appelé EDLIN.COM, comme ceci :

```
C:\DOS> DIR EDLIN.COM
Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE
Répertoire de C:\DOS
EDLIN    COM     14249 17/06/88  12:00
        1 Fichier(s)   21161984 octets libres
```

Edlin est un éditeur ligne (ce qui signifie que vous ne pouvez travailler que sur une ligne à la fois). Il vous permet de créer et de modifier des fichiers. Pour comprendre comment fonctionne EDLIN, utilisons-le pour créer un fichier de commandes, appelé DIRCOURT.BAT, qui contient les commandes CLS et DIR /W.

Pour commencer, exécutez Edlin après l'indicatif DOS et spécifiez le fichier à créer ou modifier (il est indispensable de spécifier un nom de fichier dans votre ligne de commande EDLIN). Dans ce cas, la ligne de commande est :

```
C> EDLIN DIRCOURT.BAT
```

Quand vous appuyez sur Entrée, Edlin affiche :

```
C> EDLIN DIRCOURT.BAT
Nouveau fichier
*
```

L'astérisque (symbole *) est l'indicatif d'Edlin. Edlin offre plusieurs commandes à un seul caractère. Quelques-unes de

ces commandes suffiront à nos besoins (pour une description complète d'Edlin, reportez-vous à votre manuel DOS).

Pour insérer une commande dans un fichier de traitement par lots, vous devez émettre la commande d'insertion d'Edlin, I, et appuyer sur Entrée :

```
C> EDLIN DIRCOURT.BAT
Nouveau fichier
*I
         1:*
```

Edlin vous invite à entrer la première ligne du fichier de commandes. Tapez *CLS* et appuyez sur Entrée :

```
C> EDLIN DIRCOURT.BAT
Nouveau fichier
*I
         1:* CLS
         2:*
```

Edlin vous invite maintenant à taper la deuxième ligne. Tapez *DIR/W* et appuyez sur Entrée :

```
C> EDLIN DIRCOURT.BAT
Nouveau fichier
*I
         1:* CLS
         2:* DIR /W
```

Puisque ce fichier de commandes n'exige que deux lignes, vous devez dire à Edlin que l'insertion de texte est terminée. Pour cela, maintenez la touche Contrôle (Ctrl) enfoncée et tapez C. Quand vous appuyez ainsi sur Ctrl+C, Edlin sort du mode insertion et affiche son indicatif, comme suit :

```
C> EDLIN DIRCOURT.BAT
Nouveau fichier
*I
         1:* CLS
         2:* DIR /W
         3:* ^C
*
```

Pour sauvegarder le fichier, sortir d'Edlin et revenir à DOS, entrez la commande qui met fin à Edlin, c'est-à-dire E (pour *End*). Appuyez sur Entrée :

```
C> EDLIN DIRCOURT.BAT
Nouveau fichier
*I
          1:* CLS
          2:* DIR /W
          3:* ^C
*E
C>
```

Pour exécuter le fichier de commandes que vous venez de créer, tapez *DIRCOURT* :

```
C> DIRCOURT
```

et appuyez sur Entrée. DOS efface l'écran et affiche la liste abrégée des fichiers du répertoire avec plusieurs noms de fichiers par ligne (conformément à la spécification du commutateur /W).

Comme nous l'avons vu, Edlin vous permet de changer un fichier existant. Modifions maintenant le fichier DIRCOURT.BAT pour que DOS n'affiche que les fichiers dont les noms se terminent par l'extension EXE. Comme précédemment, tapez *EDLIN* et spécifiez le nom de fichier :

```
C> EDLIN DIRCOURT.BAT
```

Puisque le fichier DIRCOURT.BAT existe déjà, Edlin affiche :

```
C> EDLIN DIRCOURT.BAT
Fin du fichier d'entrée
*
```

Dans ce cas, Edlin vous signale qu'il a lu l'ensemble du fichier et qu'il est prêt à recevoir vos modifications. Quand vous tapez le chiffre *1* après l'indicatif d'Edlin et appuyez sur Entrée, Edlin affiche la première ligne du fichier :

```
C> EDLIN DIRCOURT.BAT
Fin du fichier d'entrée
*1
          1:* CLS
          1:*
```

Par le deuxième *1:** qui apparaît à l'écran, Edlin vous invite à modifier la Ligne 1. Quand on veut laisser la ligne 1 inchangée, il suffit d'appuyer sur Entrée ; sinon, il faut taper le nouveau texte de la ligne 1. Pour l'instant, laissez la ligne inchangée (en appuyant sur Entrée) :

```
C> EDLIN DIRCOURT.BAT
Fin du fichier d'entrée
*1
          1:* CLS
          1:*
*
```

Tapez le chiffre 2 après l'indicatif d'Edlin et appuyez sur Entrée. Edlin affiche le contenu de la ligne 2, que vous pouvez changer :

```
C> EDLIN DIRCOURT.BAT
Fin du fichier d'entrée
*1
          1:* CLS
          1:*
*2
          2:* DIR /W
          2:*
```

Dans ce cas, vous devez remplacer *DIR /W*, ancien texte de la ligne 2, par le nouveau texte *DIR *.EXE /W*. Tapez la nouvelle ligne et appuyez sur Entrée :

```
C> EDLIN DIRCOURT.BAT
Fin du fichier d'entrée
*1
          1:* CLS
          1:*
*2
          2:* DIR /W
          2:* DIR *.EXE /W
*
```

En entrant à nouveau E, commande fin d'Edlin, et en appuyant sur Entrée, vous sauvegardez le contenu (mis à jour) du fichier et sortez vers DOS :

```
C> EDLIN DIRCOURT.BAT
Fin du fichier d'entrée
*1
         1:* CLS
         1:*
*2
         2:* DIR /W
         2:* DIR *.EXE /W
*E

C>
```

A présent, si vous exécutez le fichier de commandes, DOS efface l'écran et affiche uniquement les fichiers dont les noms se terminent par l'extension EXE.

Si vous voulez mettre fin à l'édition du fichier sans sauvegarder les modifications, utilisez la commande quitter d'Edlin, Q. Quand vous exécutez cette commande, Edlin affiche le message :

```
Abandon de l'édition (O/N) ?
```

Si vous tapez O (pour oui) et appuyez sur Entrée, Edlin ignore toutes les modifications, vous ramène à DOS et laisse le fichier dans l'état initial. Si vous tapez N (pour non) et appuyez sur Entrée, Edlin continue la session d'édition (modification).

Nous utiliserons Edlin, ultérieurement, pour créer des fichiers de commandes qui transforment vos réglages d'écran et pour assigner des commandes DOS d'usage courant aux touches de fonction de votre clavier. Mais tout d'abord, regardons un fichier de commandes simple que vous pourrez utiliser quotidiennement.

A l'aide d'Edlin, créons un second fichier de commandes, appelé DIRTRI.BAT, qui contient une commande permettant d'imprimer la liste triée de vos fichiers :

C> **EDLIN DIRTRI.BAT**

Nouveau fichier



*I

1:* DIR ¦ SORT > PRN

2:* ^C



*E



C>

CHAPITRE 2

Système sur mesure grâce aux fichiers de commandes

PERSONNALISER LE SYSTÈME PAR LES FICHIERS DE COMMANDES

De nombreux utilisateurs «personnalisent» leurs systèmes en modifiant l'indicatif de DOS. Par défaut, DOS affiche comme indicatif la lettre du lecteur courant suivie par le symbole plus grand que (>). Par exemple : C>.

Beaucoup de personnes souhaitent afficher également le répertoire courant dans l'indicatif. On utilise dans ce but la commande PROMPT. Si vous ouvrez le manuel d'utilisation de DOS à la commande PROMPT, vous constaterez que PROMPT reconnaît plusieurs combinaisons distinctes de caractères appelées *métacaractères.* En général, un métacaractère est un caractère placé après le symbole dollar ($). Quand la commande PROMPT détecte un métacaractère, elle affiche un texte qui correspond à ce métacaractère et à lui seul. Voici la liste des métacaractères reconnus par PROMPT :

Métacaractère	Texte correspondant
$$	Caractère $
$b	Caractère ¦
$d	Date actuelle du système
$e	Caractère ASCII Escape
$g	Caractère >

Métacaractère	Texte correspondant
$h	Caractère ASCII Retour arrière
$l	Caractère <
$n	Lecteur courant
$p	Lecteur et répertoire courants
$t	Heure actuelle du système
$v	Version courante de DOS
$_	Retour chariot, avance ligne
$q	Caractère =

OS/2 reconnaît en outre les métacaractères suivants :

Métacaractère	Texte correspondant
$a	Caractère &
$c	Caractère (
$f	Caractère)
$l	Ligne d'aide
$s	Insertion d'un espace en tête

Il est facile d'imaginer les difficultés que l'on aurait à se souvenir de ces métacaractères. Une meilleure solution consiste à déterminer le type d'indicatif que l'on désire et à mettre la commande PROMPT correspondante dans un fichier de commandes appelé MONINDIC.BAT. Par exemple, la commande ci-dessous affiche la date actuelle du système sur une ligne et, deux lignes plus bas, les lecteur et répertoire courants entre crochets :

```
PROMPT  $d$_$_[$p]
```

Dans l'hypothèse où vous placez cette commande dans le fichier de commandes MONINDIC.BAT, l'exécution de ce fichier produit des lignes d'indicatif du genre de celles-ci :

```
Mer 31/01/90

[C:\DOS]
```

Vous voyez que DOS affiche la date courante, puis le répertoire courant entre crochets (à noter que les crochets,

comme d'ailleurs tout caractère non précédé du signe $, sont affichés tels quels). Indépendamment de l'indicatif système que vous aimeriez choisir, il vous sera probablement très utile d'y inclure les lecteur et répertoire courants. En déterminant votre indicatif par un fichier de commandes, il est inutile de mémoriser d'un jour à l'autre la ligne de commande PROMPT ou de la saisir à chaque session de travail.

A la première émission de la commande PRINT, DOS installe dans la mémoire de l'ordinateur un programme qui contrôle votre imprimante et les fichiers que vous imprimez, en vous permettant d'inclure plusieurs commutateurs dans la ligne de commande (note : les commutateurs prévoyant l'addition de paramètres ne peuvent être spécifiés qu'à la première exécution de PRINT). Les commutateurs spécifient des préférences telles que l'imprimante sélectionnée (normalement PRN pour les imprimantes parallèles et COM1 pour les imprimantes série) ; l'importance de la file d'attente, de 4 à 32 (par défaut, DOS retient dix fichiers à imprimer dans une liste appelée file d'attente) ; enfin, plusieurs commutateurs déterminent le temps relatif consacré par l'ordinateur à l'impression des fichiers. Le tableau suivant décrit brièvement les commutateurs de PRINT :

Commutateur	**Fonction**
/B	Spécifie la taille du tampon d'impression
/C	Annule au moins un fichier dans la file d'attente d'impression
/D	Spécifie le périphérique de sortie visé
/M	Spécifie le nombre d'impulsions d'horloge pendant lequel PRINT garde le contrôle de l'imprimante à chaque fois qu'elle commence à imprimer
/Q	Spécifie le nombre de fichiers que la file d'attente d'impression peut stocker
/S	Spécifie la tranche de temps réservée à l'impression (fixe la fréquence de prise de contrôle de l'imprimante par PRINT)

Commutateur	Fonction
/T	Annule tous les fichiers dans la file d'attente d'impression
/U	Spécifie le nombre d'impulsions d'horloge pendant lequel PRINT peut attendre la disponibilité de l'imprimante
/P	Imprime les fichiers mentionnés sur cette ligne de commande

La plupart des utilisateurs se contentent des réglages par défaut, tout simplement parce qu'ils ont oublié les commutateurs mis à leur disposition. Dans de nombreux cas, on peut cependant obtenir de meilleures performances de l'imprimante et de l'ordinateur en incluant quelques commutateurs simples dans la commande PRINT. Les fichiers de commandes DOS vous donnent un moyen aisé d'exécuter PRINT avec ses régages optimaux. La ligne de commande ci-dessous donnera les meilleurs résultats pour la grande majorité des utilisateurs :

```
PRINT /D:PRN /Q:32 /M:64 /U:16 /B:4096
```

Si vous utilisez une imprimante série connectée au port COM1:, modifiez le commutateur /D en conséquence :

```
PRINT /D:COM1 /Q:32 /M:64 /U:16 /B:4096
```

Dans les deux cas, la commande installe une file d'attente pouvant admettre jusqu'à trente-deux fichiers, ceci grâce au commutateur /Q. Le commutateur /D indique à PRINT l'imprimante destinataire. Lorsqu'elle comprend ce commutateur, la commande PRINT supprime le message (*Nom de l'imprimante [PRN]:*) que PRINT affiche normalement à la première utilisation. Le commutateur /M demande à PRINT de garder le contrôle de l'ordinateur pendant 64 impulsions d'horloge à chaque démarrage de l'impression. Le commutateur /U impose à PRINT de retenir le contrôle pendant 16 impulsions d'horloge lorsqu'il attend que l'imprimante redevienne disponible, si elle est actuellement occupée.

Finalement, le commutateur /B réserve une mémoire tampon de 4096 octets.

PRINT remplit ce tampon avec des caractères du fichier chaque fois qu'il effectue une lecture sur disque. La mémoire tampon choisie étant importante (comparée aux 512 octets par défaut), PRINT réduit le nombre d'opérations lentes d'accès au disque qu'il doit exécuter pour imprimer un fichier. La performance globale du système s'en trouve améliorée.

Le fichier de commandes pourrait recevoir l'appellation INSTIMP.BAT (pour Installer Imprimante). Bien que très simples, MONINDIC.BAT et INSTIMP.BAT vous montrent comment les fichiers de commandes éliminent la nécessité de mémoriser des commandes difficiles.

Vous pourriez envisager l'exécution de ces commandes sur une base quotidienne ou même à chaque lancement de DOS. En fait, DOS vous permet de créer un fichier de commandes spécial appelé AUTOEXEC.BAT, dont le contenu est exécuté par DOS à chaque mise sous tension ou réamorçage à chaud du système. Si vous utilisez OS/2, le système d'exploitation exécute un fichier de commandes similaire dont le nom est STARTUP.CMD, à chaque démarrage de l'ordinateur. AUTOEXEC.BAT et STARTUP.CMD ont la même fonction : chacun d'eux offre un jeu de commandes qui s'exécutent automatiquement à chaque lancement du système.

EXAMINER LE FICHIER AUTOEXEC.BAT

L'utilisateur a généralement besoin d'exécuter plusieurs commandes DOS au début de chaque session de travail. Ces commandes incluent PROMPT (qui détermine l'indicatif du système), PRINT (qui installe le programme de gestion de l'imprimante), PATH (qui définit le jeu de lecteurs et répertoires où DOS doit rechercher les programmes exécutables), APPEND (qui définit le chemin de recherche des fichiers de données) et SET (qui détermine les variables d'environnement de DOS), sans oublier les commandes

étrangères à DOS qui installent des programmes résidant en mémoire. Si vous groupez toutes ces commandes dans un même fichier de traitement par lots, vous n'aurez pas besoin de les frapper à chaque lancement du système.

Lorsque DOS démarre, il cherche dans le répertoire racine (sommet de la hiérarchie) du disque de démarrage un fichier ayant pour nom AUTOEXEC.BAT. Si ce fichier existe, DOS l'ouvre et exécute les commandes qu'il contient, de la première à la dernière.

S'il ne trouve pas de fichier AUTOEXEC.BAT dans le répertoire racine, DOS affiche un message de copyright et exécute les commandes DATE et TIME. Pour mieux saisir ce processus, choisissons un exemple particulier de fichier AUTOEXEC.BAT :

```
PATH C:\DOS
PROMPT $p$g
PRINT /D:LPT1 /Q:32 /M:64 /U:16 /B:4096
```

Ce fichier contient trois commandes que l'on trouve souvent dans AUTOEXEC.BAT. La première commande :

```
PATH C:\DOS
```

définit le chemin de recherche des commandes. Chaque fois que DOS ne parvient pas à trouver une commande dans le répertoire courant ou dans le répertoire spécifié, il vérifie si vous avez défini un chemin de recherche. La commande PATH de DOS vous permet de spécifier une liste d'un ou plusieurs sous-répertoires dans lesquels il recherchera les programmes exécutables. Dans le cas de notre exemple, si DOS ne trouve pas le programme spécifié, il le recherche dans le sous-répertoire DOS du lecteur C. Puisque ce sous-répertoire contient toutes les commandes externes de DOS, il est fort probable que DOS parviendra à localiser le programme. Si votre disque a d'autres sous-répertoires qui contiennent des fichiers d'usage courant, PATH vous permet de les inclure dans le chemin de recherche du fichier de

commandes, en séparant les répertoires par des points-virgules, comme ceci :

```
PATH C:\DOS;C:\UTIL;C:\BIN
```

Dans ce cas, si DOS ne réussit pas à localiser un programme dans le répertoire courant ou dans le répertoire spécifié, il recherche le programme dans le sous-répertoire C:\DOS. Si DOS trouve le programme dans C:\DOS, il l'exécute. Dans le cas contraire, la recherche continue dans les répertoires C:\UTIL et C:\BIN (dans cet ordre).

Les seuls répertoires à mentionner dans la commande PATH sont ceux qui ont toutes les chances de contenir des programmes exécutables. Chaque fois que DOS utilise le chemin de recherche de cette commande, il examine chaque fichier dans chaque répertoire spécifié par le chemin, pour s'arrêter quand il trouve le fichier. Si un sous-répertoire n'a que très peu de chances de contenir un programme exécutable, DOS perd du temps à passer en revue les fichiers de ce sous-répertoire.

La deuxième commande de notre fichier :

```
PROMPT $p$g
```

détermine un indicatif DOS composé des lecteur et répertoire courants, suivis du symbole >. Chaque fois que vous changez de lecteur ou de répertoire, DOS modifie l'indicatif pour afficher les lecteur et répertoire courants par défaut. Dans notre exemple, en admettant que le répertoire courant est le répertoire racine du lecteur C, votre indicatif est :

```
C:\>
```

Si vous utilisez CHDIR (Change Directory) pour choisir, par exemple, le sous-répertoire DOS, l'indicatif change comme ci-après :

```
C:\> CHDIR \DOS
C:\DOS>
```

Si vous changez de lecteur courant, DOS change également l'indicatif. Par exemple, voici ce que devient l'indicatif quand vous passez du lecteur C au lecteur A :

```
C:\DOS> A:
A:\>
```

Pour des raisons pratiques, la plupart des utilisateurs affichent les lecteur et répertoire courants dans leur indicatif.

La troisième commande :

```
PRINT /D:LPT1 /Q:32 /M:64 /U:16 /B:4096
```

installe le programme utilisé par DOS pour gérer les fichiers que vous imprimerez ultérieurement à l'aide de la commande PRINT.

Les commandes que votre fichier AUTOEXEC.BAT contient effectivement peuvent varier en fonction de vos programmes. Si votre système ne comporte pas d'horloge alimentée par pile ou accumulateur pour stocker la date et l'heure réelles, vous devez inclure les commandes DOS DATE et TIME dans AUTOEXEC.BAT. Ainsi, vous serez invité à entrer les données de mise à l'heure et à la date correctes, chaque fois que vous lancerez l'ordinateur.

Soyez particulièrement attentif lorsque vous changez le contenu du fichier AUTOEXEC.BAT. Il est bon de suivre la règle suivante : vous ne modifiez jamais AUTOEXEC.BAT sans avoir, au préalable, pris une copie de secours du fichier en utilisant la commande :

```
C> COPY AUTOEXEC.BAT AUTOEXEC.SAV
```

Si un jour vous voulez accéder à la version précédente du fichier, elle est immédiatement disponible sur votre disque.

AUTOEXEC.BAT ET CONFIG.SYS

Beaucoup d'utilisateurs de DOS ne perçoivent pas très clairement la différence entre AUTOEXEC.BAT et

CONFIG.SYS. AUTOEXEC.BAT est un fichier de traitement par lots dont les commandes sont exécutées par DOS à chaque lancement du système. En revanche, CONFIG.SYS n'est pas un fichier de traitement par lots (souvenez-vous : les fichiers de commandes doivent comporter l'extension .BAT). Les fichiers qui comportent l'extension .SYS sont des fichiers du système d'exploitation qui accomplissent des tâches bien spécifiques. Dans le cas de CONFIG.SYS, le fichier contient des valeurs que DOS utilise pour s'autoconfigurer en mémoire à chaque démarrage du système. Ces valeurs incluent, par exemple, l'entrée BUFFERS= (qui définit le nombre de mémoires tampons d'entrées/sorties disque utilisées par DOS) et l'entrée FILES= (qui spécifie le nombre maximal de fichiers ouverts simultanément). Le tableau suivant décrit brièvement les entrées du fichier CONFIG.SYS :

Entrée	Fonction
BREAK=	Met En/Hors service la vérification complète de la commande d'arrêt Ctrl+C
BUFFERS=	Spécifie le nombre des zones tampons d'entrées/sorties disque
COUNTRY=	Identifie un nouveau pays, si les informations spécifiques du pays sont disponibles (apparu avec MSD-DOS 2.1 et PC-DOS 3.0)
DEVICE=	Installe un pilote de périphérique
DRIVPARM=	Spécifie les caractéristiques des périphériques travaillant par blocs (particulier à MS-DOS, à partir de 3.2)
FCBS=	Spécifie le nombre de blocs de contrôle de fichier reconnus par DOS pour les programmes anciens (à partir de la version 3.0)
FILES=	Spécifie le nombre maximal de fichiers ouverts simultanément par DOS
INSTALL=	Permet de charger certaines commandes DOS pendant l'exécution de CONFIG.SYS (à partir de la version 4.0)
LASTDRIVE=	Spécifie la lettre d'identification du dernier lecteur accessible par DOS (à partir de la version 3.0)

Entrée	Fonction
REM=	Permet de mettre des lignes de commentaires dans un fichier CONFIG.SYS (à partir de 4.0)
SHELL=	Définit un interpréteur de commandes autre que DOS
STACKS=	Fournit un espace de pile supplémentaire pour les systèmes qui rencontrent trop d'interruptions à un instant donné (MS-DOS à partir de 3.2, PC-DOS à partir de 3.3)
SWITCHES=	Utilise les fonctions du clavier classique lorsqu'un clavier étendu est installé (propre à PC-DOS 4.0)

A chaque démarrage, DOS s'installe dans la mémoire de l'ordinateur avant d'exécuter toute autre commande. Pendant ce processus, DOS examine le contenu du fichier CONFIG.SYS dont il utilise les entrées pour personnaliser le système d'exploitation. Si ce fichier n'existe pas, DOS utilise des valeurs par défaut.

C'est seulement après son installation complète que DOS vérifie l'existence du fichier AUTOEXEC.BAT. Ainsi, bien qu'AUTOEXEC.BAT et CONFIG.SYS soient tous deux exploités par DOS durant la phase de démarrage, ces deux fichiers ont des attributions distinctes et sont appelés dans un ordre bien défini.

CHAPITRE 3

Commandes et concepts du traitement par lots

SUPPRIMER L'AFFICHAGE DES NOMS DE COMMANDES

Par défaut, chaque fois que vous lancez un fichier de commandes, DOS affiche le nom de chacune des commandes dès qu'il l'exécute. Reprenons le fichier TOUSTOC.BAT, qui nous a déjà servi d'exemple et qui contient les commandes CALCSTOC, TRISTOC, IMPSTOC et ACHASTOC.

Vous vous en souvenez peut-être, DOS affiche le nom de ces commandes au fur et à mesure de leur exécution. Il en résulte que votre écran contient l'affichage suivant :

```
C> TOUSTOC

C> CALCSTOC

C> TRISTOC

C> IMPSTOC

C> ACHASTOC

C>
```

Vous ne souhaiterez pas toujours afficher les noms de ces commandes lors de leur exécution, et cela pour diverses

raisons (réduire l'encombrement de l'écran, la confusion de l'opérateur, etc.). Selon votre version de DOS, vous utiliserez l'une ou l'autre des deux méthodes ci-après pour supprimer l'affichage des noms de commandes. Pour les utilisateurs d'OS/2 ou de DOS à partir de la version 3.3, vous mettez simplement le caractère @ en tête du nom de commande. Vous donnez ainsi à DOS l'ordre de supprimer l'affichage du nom de commande. Par exemple, le fichier VERVOL.BAT affiche le numéro de version de DOS ainsi que les nom et numéro de série du volume correspondant au disque courant.

```
@VER
@VOL
```

Comme les deux commandes @VER et @VOL contiennent le caractère @, DOS n'affiche pas le nom de commande. Quand vous exécutez ce fichier de commandes, DOS affiche seulement la version de DOS et les informations sur le volume, comme ceci :

```
MS-DOS Version 4.00

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE

C>
```

Comparez ces informations à celles que produit le fichier VERVOL2.BAT dont les deux commandes (VER et VOL) ne suppriment pas l'affichage des noms de commandes. Quand vous exécutez ce fichier de commandes, l'affichage produit par DOS est le suivant :

```
C> VER

MS-DOS Version 4.00

C> VOL

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE

C>
```

Rappelez-vous : pour utiliser le caractère @ dans un fichier de commandes, vous devez disposer de DOS (à partir de la version 3.3) ou d'OS/2.

Pour les utilisateurs d'une version plus ancienne de DOS, la commande de traitement par lots ECHO permet à votre fichier de supprimer l'affichage du nom de commande. Avec ces versions plus anciennes de DOS, on met fréquemment ECHO OFF en position de première commande du fichier de traitement par lots. Si DOS rencontre ECHO OFF, il n'affichera aucun des noms des commandes suivantes du fichier lors de leur exécution. Par exemple, le fichier de commandes VERVOL3.BAT utilise ECHO OFF, comme ci-dessous :

```
ECHO OFF
VER
VOL
```

Quand vous lancez ce fichier de commandes, DOS affiche :

```
C> ECHO OFF

MS-DOS Version 4.00

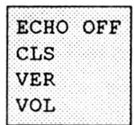
Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE

C>
```

Vous constatez que DOS n'a pas affiché les noms de commandes VER et VOL. En revanche, il a affiché le message :

```
C> ECHO OFF
```

Si vous utilisez DOS version 3.3 ou postérieure ou OS/2, vous pouvez éliminer le message ECHO OFF en précédant la commande ECHO OFF du caractère @. Si vous utilisez une version plus ancienne de DOS, vous pouvez mettre la commande CLS dans le fichier de commandes, immédiatement après ECHO OFF (comme l'indique la suite de commandes ci-après), pour effacer l'écran avant d'afficher la sortie des commandes VER et VOL.



```
ECHO OFF
CLS
VER
VOL
```

Supposons que le fichier de commandes ci-dessus ait pour nom VERVOL4.BAT. Quand vous l'exécutez, DOS affiche :

```
MS-DOS Version 4.00

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE

C>
```

Plus loin, nous ferons largement appel à la commande ECHO dans nos fichiers de commandes pour afficher des messages à l'opérateur, pour déterminer les couleurs de l'écran et même pour redéfinir les touches du clavier. Pour le moment, retenez surtout que la commande ECHO OFF inhibe l'affichage des commandes d'un fichier de traitement par lots pendant l'exécution de ce fichier par DOS. Si vous avez une raison précise d'afficher certaines commandes pendant l'exécution (nous en verrons des exemples), la commande ECHO ON active l'affichage du nom de commande (mode par défaut de DOS). Cette dernière version, appelée VERVOL5.BAT, du fichier de commandes VER, VOL, illustre l'emploi d'ECHO OFF et ECHO ON :

```
ECHO OFF
CLS
VER
ECHO ON
VOL
```

Si vous exécutez ce fichier de commandes, DOS affiche :

```
MS-DOS Version 4.00

C> VOL

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE

C>
```

Vous voyez que DOS a supprimé l'affichage du nom de commmande pour CLS, VER et ECHO ON. Lorsque l'exécution d'ECHO ON a été terminée, DOS a réactivé l'affichage du nom de commande. On lit donc le nom de la commande VOL.

Les utilisateurs préfèrent souvent précéder chacune des commandes de leur fichier AUTOEXEC.BAT par le caractère @ ou la commande ECHO OFF. Dans les deux cas, DOS supprime l'affichage des noms de commandes pendant qu'il exécute le fichier AUTOEXEC.BAT.

Commande de traitement par lots ECHO

Fonction :

Supprime ou active l'affichage des noms de commandes pendant l'exécution par DOS d'un fichier de traitement par lots.

Format :

ECHO ON

ou

ECHO OFF

Remarques :

La commande ECHO OFF supprime l'affichage du nom des commandes d'un fichier de traitement par lots lorsque DOS exécute ces commandes. En supprimant l'affichage des noms de commandes, vous réduisez l'encombrement de l'écran et l'embarras éventuel de l'opérateur.

Par défaut, DOS utilise ECHO ON, qui prescrit l'affichage systématique des commandes exécutées. DOS réactive ECHO ON à la fin de chaque fichier de commandes.

(suite)

Si vous utilisez DOS version 3.3 (et au-delà) ou OS/2, vous pouvez supprimer le nom de chaque commande d'un fichier de traitement par lots en précédant ce nom du caractère @.

Outre qu'elle active ou désactive l'affichage des noms de commandes, la commande de traitement par lots ECHO permet à vos fichiers de commandes d'afficher des messages destinés à l'opérateur.

Exemple

Hypothèse : un fichier de traitement par lots contient les commandes DOS TIME et DATE, comme ci-dessous :

```
TIME
DATE
```

Quand vous exécutez le fichier de commandes, DOS (par défaut) affiche le nom de chaque commande.

```
C> TIME
L'heure courante est : 10:23:17,45
Donnez la nouvelle heure :

C> DATE
La date courante est : Lun 29/01/90
Donnez la nouvelle date (jj/mm/aa) :
```

Si vous ajoutez les commandes DOS ECHO OFF et CLS à votre fichier de traitement par lots, comme ci-dessous :

```
ECHO OFF
CLS
TIME
DATE
```

DOS supprime l'affichage des noms de commandes :

```
L'heure courante est : 10:23:17,45
Donnez la nouvelle heure :
La date courante est : Lun 29/01/90
Donnez la nouvelle date (jj/mm/aa) :
```

INTERROMPRE L'EXÉCUTION D'UN FICHIER DE COMMANDES

Si vous souhaitez mettre fin à l'exécution d'un fichier de commandes, maintenez la touche Ctrl enfoncée et appuyez sur C. Quand DOS détecte l'ordre Ctrl+C, il affiche le message suivant :

```
Arrêt du traitement par lots (O/N) ?
```

Si vous choisissez O, DOS interrompt immédiatement l'exécution du traitement par lots. Si vous tapez N, DOS met fin à l'exécution de la commande DOS en cours et reprend l'exécution à la commande suivante du fichier. Pour mieux comprendre ce processus, examinons une fois de plus le fichier de commandes TIMEDATE.BAT, qui contient TIME et DATE. Quand vous lancez ce fichier de commandes, DOS exécute d'abord la commande TIME, comme ceci :

```
C> TIMEDATE

L'heure courante est : 10:23:17,45
Donnez la nouvelle heure :
```

Au lieu d'entrer une nouvelle heure, utilisez la combinaison de touches Ctrl+C. Lorsque DOS détecte l'ordre Ctrl+C, l'écran affiche le message vous demandant si vous désirez mettre fin à l'exécution du fichier de commandes, comme ci-dessous :

```
C> TIMEDATE

L'heure courante est : 10:23:17,45
Donnez la nouvelle heure : ^C

Arrêt du traitement par lots (O/N) ?
```

Dans ce cas, appuyez sur O. Ainsi, vous mettez fin à l'exécution de l'ensemble du fichier de commandes et redonnez le contrôle à l'indicatif DOS.

Exécutez à nouveau TIMEDATE.BAT et apppuyez sur Ctrl+C après le message d'invite de la commande TIME :

```
C> TIMEDATE

L'heure courante est : 10:23:17,45
Donnez la nouvelle heure : ^C

Arrêt du traitement par lots (O/N) ?
```

Cette fois, appuyez sur N. Quand vous appuyez sur N, DOS met fin à l'exécution non pas du fichier de commandes, mais seulement de la commande en cours. L'exécution du fichier de commandes reprend à la commande suivante, qui dans notre exemple est DATE :

```
C> TIMEDATE

L'heure courante est : 10:23:17,45
Donnez la nouvelle heure : ^C

Arrêt du traitement par lots (O/N) ? N

La date courante est : Lun 29/01/90
Donnez la nouvelle date (jj/mm/aa) :
```

Remarque : avec OS/2, Ctrl+C n'offre pas de choix, met fin à l'exécution du fichier de commandes et redonne immédiatement le contrôle à l'indicatif du système.

REDIRIGER LA SORTIE PAR DES FICHIERS DE COMMANDES DOS

La redirection des entrées/sorties de commandes travaille non pas *pour* le fichier de traitement par lots que vous exécutez, mais *dans* le fichier. Par exemple, si un fichier de commandes appelé DIRAB.BAT se compose de :

```
DIR A:*.*
DIR B:*.*
```

et si vous exécutez ce fichier en tapant :

```
DIRAB > PRN
```

la sortie DIR ne *sera pas* dirigée vers l'imprimante. Pour diriger la sortie DIR vers l'imprimante, la redirection des

entrées/sorties doit être associée directement à une commande DOS spécifique. Par suite, DIRAB.BAT devrait être récrit de la façon suivante :

```
DIR A:*.* > PRN
DIR B:*.* > PRN
```

Dans ce cas, DOS redirigera la sortie des deux commandes DIR vers l'imprimante. Le fichier de commandes affichera encore les deux lignes de commande DIR à l'écran lors de leur exécution. Pour supprimer l'affichage des noms de commandes, utilisez la commande ECHO OFF ou le caractère @ placé de la façon suivante :

```
@DIR A:*.* > PRN
@DIR A:*.* > PRN
```

FACILITER L'INTERPRÉTATION DES FICHIERS DE COMMANDES

Le nom que vous donnez à un fichier de commandes devrait suggérer le rôle global de ce fichier. Cependant, quand la complexité de vos fichiers de commandes va s'accroître, vous aurez de plus en plus de difficultés à vous souvenir non seulement de la séquence de commandes du fichier, mais encore des commutateurs que contient chaque ligne de commande. Pour vous aider à vous rappeler la fonction d'un fichier de commandes ou pour aider une autre personne à comprendre cette fonction en lisant le fichier, DOS met à votre disposition la commande de traitement par lots REM (pour Remarque). Avec REM vous incorporez à votre fichier de commandes des informations destinées à tout utilisateur lisant ce fichier. Quand DOS rencontre REM, l'exécution continue à la commande suivante du fichier de traitement par lots.

Vous constaterez ci-après que la commande REM améliore la lisibilité du fichier de commandes TOUSTOC.BAT :

```
ECHO OFF
REM Nom : TOUSTOC.BAT
REM Fonction: Exécute les commandes de
REM traitement du stock
REM
REM Ecrit par K. Jamsa 01/06/90
REM
REM Le programme CALCSTOC détermine
REM la situation actuelle du stock.
CALCSTOC
REM Le programme TRISTOC produit une
REM liste triée du stock actuel.
TRISTOC
REM Le programme IMPSTOC imprime les
REM états sur papier du stock actuel.
IMPSTOC
REM Le programme ACHASTOC établit les
REM commandes de réapprovisionnement.
ACHASTOC
```

A première vue, la longueur du fichier de commandes peut vous sembler impressionnante. Pourtant, la lecture des remarques devrait clarifier pour vous le fonctionnement de ce fichier.

La commande ECHO OFF désactive l'affichage des noms de commandes lors de l'exécution du fichier. Si vous n'incluez pas la commande ECHO OFF, DOS affichera chaque commande REM à l'écran pendant l'exécution (souvenez-vous que REM est destiné à vous aider à interpréter le fichier, mais l'utilisateur final serait perturbé par un affichage systématique de toutes les remarques pendant l'exécution du fichier de commandes).

Ensuite, les lignes 2 à 6 expliquent le rôle du fichier de commandes, indiquent le nom de l'auteur et la date de création. En incluant ces informations, vous donnez un point de contact à l'utilisateur éventuel du fichier de commandes et vous le renseignez sur l'âge du fichier.

Toute personne décidant de modifier le fichier devrait incorporer une ligne mentionnant la date et la raison de cette modification.

```
REM Ecrit par K. Jamsa  06/01/90
REM Dernière modification : D. Jamsa 12/09/90
REM Impression 2 copies de l'état du stock.
REM
```

Plusieurs semaines après sa création, le plus simple des fichiers de commandes vous paraîtra obscur. Les quelques minutes passées aujourd'hui à le documenter vous feront gagner plus tard beaucoup de temps et d'efforts.

Commande de traitement par lots REM

Fonction :

Introduit dans un fichier de commandes des remarques expliquant le fonctionnement de ce fichier.

Format :

REM texte

Remarques :

Les difficultés d'interprétation d'un fichier de commandes croissent avec sa complexité. La commande REM permet d'inclure dans un fichier de commandes un aide-mémoire expliquant le rôle de ce fichier. Quand DOS exécute le fichier de commandes, il ignore l'instruction REM et continue à la commande suivante.

La commande ECHO OFF supprime l'affichage des remarques lorsqu'on exécute le fichier de commandes.

Exemple

Les remarques indiquent le rôle du fichier de commandes, le nom de l'auteur et la date de création.

```
REM Procédures mensuelles de sauvegarde.
REM Ecrit par : K. Jamsa
REM             06/01/90
REM Fonction : Effectue une copie de secours
REM            de vos fichiers sur disque.
```

AMÉLIORER LA LISIBILITÉ DES FICHIERS DE COMMANDES

La commande de traitement par lots REM vous permet d'insérer dans un fichier de commandes des remarques significatives auxquelles vous pouvez ensuite vous référer pour intrerpréter le traitement qu'effectue ce fichier. Quand vous augmentez le nombre de ces remarques, la probabilité d'encombrement s'accroît elle aussi, ce qui peut conduire à créer la confusion dans l'esprit de l'utilisateur.

Pour clarifier l'aspect du fichier de commandes, les utilisateurs laissent souvent des lignes blanches pour distinguer des groupes de commandes associées logiquement. On attire ainsi l'attention du lecteur sur des sections particulières du fichier. Un fichier de commandes composé de sections plus petites est moins intimidant pour le lecteur. En outre, grâce aux lignes blanches, l'apparence visuelle du fichier de commandes est plus agréable :

```
ECHO OFF
REM Nom : TOUSTOC.BAT
REM Fonction : Exécute les commandes de
REM traitement du stock
REM
REM Ecrit par K. Jamsa 01/06/90
REM

REM Le programme CALCSTOC détermine
REM la situation actuelle du stock.
CALCSTOC

REM Le programme TRISTOC produit une
REM liste triée du stock actuel.
TRISTOC

REM Le programme IMPSTOC imprime les
REM états sur papier du stock actuel.
IMPSTOC

```

```
REM Le programme ACHASTOC établit les
REM commandes de réapprovisionnement.
ACHASTOC
```

SUSPENDRE LE TRAITEMENT PAR LOTS

Dans la majorité des cas, les fichiers de traitement par lots exécutent toutes les commandes, de la première à la dernière, sans intervention de l'utilisateur. Cependant, un fichier de commandes est parfois conduit à attendre de l'utilisateur qu'il mette une imprimante sur la position «en ligne» ou qu'il insère une certaine disquette dans un lecteur. Dans de tels cas, la commande de traitement par lots PAUSE donne au fichier de commandes le moyen d'afficher un message qui suspend le traitement jusqu'à ce que l'utilisateur appuie sur une touche quelconque pour continuer.

Quand un fichier de commandes exécute la commande PAUSE, le message spécifié s'affiche, suivi de la ligne :

```
Appuyez sur une touche pour continuer . . .
```

Dès que l'utilisateur appuie sur une touche quelconque, le fichier de commandes reprend le traitement à la commande suivante. Si pour une raison quelconque, l'utilisateur désire mettre fin au traitement, il peut appuyer sur Ctrl+C en réponse au message de PAUSE.

Considérons un fichier de commandes IMDIR.BAT qui, avant d'imprimer la liste des fichiers du répertoire courant, affiche les lignes suivantes :

```
PAUSE Mettez l'imprimante ON LINE
Appuyez sur une touche pour continuer . . .
```

Vous l'avez probablement deviné, le fichier de commandes contient les deux commandes ci-dessous :

```
PAUSE Mettez l'imprimante ON LINE
DIR > PRN
```

Quand vous lancez ce fichier de traitement par lots, PAUSE attend que vous ayez appuyé sur une touche pour continuer.

Le fichier de commandes DSKIMDIR.BAT emploie deux commandes PAUSE pour obtenir une liste imprimée des fichiers contenus dans le lecteur A. La première PAUSE invite l'utilisateur à insérer une disquette dans le lecteur A. La deuxième lui demande de mettre l'imprimante en ligne.

```
PAUSE Mettez la disquette dans le lecteur A
PAUSE Mettez l'imprimante ON LINE
DIR > PRN
```

Quand vous lancez ce fichier, DOS exécute la première commande PAUSE et affiche :

```
PAUSE  Mettez la disquette dans le lecteur A
Appuyez sur une touche pour continuer . . .
```

Quand vous insérez une disquette dans le lecteur A et appuyez sur une touche, le fichier de traitement par lots exécute la seconde commande PAUSE et affiche :

```
PAUSE  Mettez l'imprimante ON LINE
Appuyez sur une touche pour continuer . . .
```

Quand vous mettez l'imprimante en ligne et appuyez sur une touche, le fichier de commandes exécute DIR et redirige la sortie vers l'imprimante, comme prévu.

Vous noterez qu'aucun des deux fichiers ci-dessus n'utilise la commande ECHO OFF pour supprimer l'affichage des noms de commandes. Il se trouve en effet qu'ECHO OFF supprime le message à l'utilisateur contenu dans la ligne de commande PAUSE. Il en résulte qu'avec ECHO OFF , la seule sortie qui apparaîtrait à l'écran serait :

```
Appuyez sur une touche pour continuer . . .
```

Le fichier de commandes suivant, appelé SUPPRMSG.BAT, illustre la suppression à l'écran du message de la ligne de commande PAUSE :

```
ECHO OFF
PAUSE Mettez l'imprimante ON LINE
DIR > PRN
```

Quand vous exécutez ce fichier de commandes, DOS produit l'affichage suivant :

```
C> ECHO OFF
Appuyez sur une touche pour continuer . . .
```

Vous voyez qu'en effet, DOS a bien supprimé le message *Mettez l'imprimante ON LINE.*

La plupart du temps, un fichier de traitement par lots qui utilise la commande PAUSE fera appel à la commande ECHO ON pour activer l'affichage de certains noms de commandes et la commande ECHO OFF pour en désactiver d'autres.

Considérons le fichier de commandes ECHOTEST.BAT qui active et désactive l'affichage des noms de commandes de telle sorte que le message de PAUSE ne soit pas supprimé :

```
ECHO OFF
VER
VOL
ECHO ON
PAUSE Commande ECHO vient d'être activée
ECHO OFF
VER
VOL
```

Quand vous exécutez ce fichier, DOS affiche :

```
C> ECHOTEST

C> ECHO OFF

MS-DOS Version 4.00

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est OF35-18EE

C> PAUSE Commande ECHO vient d'être activée
Appuyez sur une touche pour continuer . . .

C> ECHO OFF

MS-DOS Version 4.00

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est OF35-18EE

C>
```

Rappelez-vous : si vous utilisez Dos version 3.3 (ou postérieure) ou OS/2, vous pouvez supprimer ECHO OFF, avec @ comme premier caractère de la ligne de commande.

Au moins pour les fichiers dont le nombre de commandes ou le temps d'exécution atteint une certaine importance, il est très probable que l'on va lancer le fichier de commandes et délaisser l'ordinateur pour passer à d'autres tâches. Si le fichier exécute une commande PAUSE, un temps considérable risque de s'écouler avant que l'utilisateur ne se souvienne qu'il doit intervenir et appuyer sur une touche pour continuer le traitement par lots.

Vous pouvez très bien demander à PAUSE, non seulement d'afficher un message à l'opérateur, mais encore de faire émettre un signal sonore par le haut-parleur incorporé à votre unité centrale. Quelques petits secrets sur le jeu de caractères ASCII (utilisé par l'ordinateur pour afficher lettres, chiffres et symboles) facilitent l'élaboration d'un signal sonore. Nous allons les découvrir.

Chaque lettre ou chiffre visualisé ou imprimé est représenté par un nombre unique compris entre 0 et 127. L'ensemble de ces nombres constitue le jeu de caractères ASCII. En plus des chiffres, des lettres majuscules et minuscules, le jeu ASCII contient des valeurs correspondant aux signes de ponctuation courants, et aussi des valeurs spéciales ayant pour l'ordinateur une signification bien précise. La valeur ASCII 7, par exemple, donne à l'ordinateur l'ordre d'envoyer un signal basse fréquence au haut-parleur interne.

Pour que PAUSE fasse retentir la «sonnerie» de l'ordinateur, la ligne de commande PAUSE doit contenir le caractère ASCII 7. Il existe plusieurs méthodes pour inclure ce caractère.

Commençons par créer un fichier de commandes que nous appellerons SON.BAT, par copie à partir du clavier :

```
C> COPY CON SON.BAT
```

Tapez le mot *PAUSE* suivi d'un espace, mais n'appuyez pas sur Entrée. Mettez ensuite trois caractères ASCII sur la ligne de commande PAUSE. Ainsi, la sonnerie retentira trois fois quand vous exécuterez le fichier de commandes.

Trois méthodes sont disponibles pour saisir un caractère ASCII 7. La première consiste à maintenir la touche Alt enfoncée et taper le chiffre 7 sur le pavé numérique à droite de votre clavier (notez bien qu' il est indispensable d'utiliser le pavé numérique, et non la rangée de chiffres en haut du clavier).

Le pavé numérique (en grisé) sur le clavier 83 touches.

Quand vous relâchez la touche Alt, DOS affiche les caractères ^G (prononcez «Contrôle G»), comme ceci :

```
C> COPY CON SON.BAT
PAUSE ^G
```

Appuyez encore deux fois sur la combinaison de touches Alt+7 (n'oubliez pas de relâcher la touche Alt entre deux entrées successives de valeurs ASCII).

```
C> COPY CON SON.BAT
PAUSE ^G^G^G
```

Terminez le fichier de commandes en entrant le message *SONNE SONNE SONNE* :

```
C> COPY CON SON.BAT
PAUSE ^G^G^GSONNE SONNE SONNE
^Z
     1 Fichier(s) copié(s)

C>
```

Lorsque vous exécutez le fichier de commandes, DOS fait entendre la sonnerie trois fois, affiche le message *Appuyez sur une touche pour continuer . . .* et attend votre intervention.

DOS et OS/2 représentent le caractère ASCII 7 à l'écran par ^G. La seconde méthode de création d'un fichier de commandes activant le signal sonore utilise la combinaison de touches Ctrl+G.

Comme précédemment, créez le fichier de commandes SON.BAT par une opération de copie à partir du clavier. Dans ce cas, nous allons simplement récrire le fichier sur la version précédente stockée sur disque. Tapez le mot *PAUSE*, suivi d'un espace :

```
C> COPY CON SON.BAT
PAUSE
```

Maintenez la touche Contrôle (Ctrl) enfoncée et appuyez sur la touche G. Dos affiche :

```
C> COPY CON SON.BAT
PAUSE ^G
```

Répétez deux fois ce processus et tapez trois fois le mot *SONNE*. Vous créez le fichier ci-dessous :

```
C> COPY CON SON.BAT
PAUSE  ^G^G^GSONNE SONNE SONNE
^Z
     1 Fichier(s) copié(s)

C>
```

Si vous exécutez ce fichier de commandes, DOS fait entendre trois fois le signal sonore de l'ordinateur, affiche les mots *SONNE SONNE SONNE*, affiche le message *Appuyez sur une touche pour continuer . . .* et attend votre réaction.

Détruisez maintenant le fichier SON.BAT, que vous allez recréer en utilisant la commande EDLIN de DOS :

```
C> DEL SON.BAT
```

La troisième méthode utilisable pour activer le signal sonore crée le fichier SON.BAT à l'aide de la commande DOS EDLIN :

```
C> EDLIN SON.BAT
Nouveau fichier
*
```

Pour entrer la première ligne du fichier, choisissez la commande insertion d'Edlin en appuyant sur la touche I. Tapez le mot *PAUSE* suivi d'un espace :

```
C> EDLIN SON.BAT
Nouveau fichier
*I
        1:* PAUSE
```

Appuyez trois fois sur Ctrl+G pour placer le caractère ASCII 7 dans le fichier et tapez trois fois le mot *SONNE*.

```
C> EDLIN SON.BAT
Nouveau fichier
*I
        1:* PAUSE ^G^G^GSONNE SONNE SONNE
        2:*
```

Appuyez sur Ctrl+C pour quitter le mode insertion d'Edlin et tapez E, commande fin d'Edlin, pour sauvegarder le contenu du fichier :

```
C> EDLIN SON.BAT
Nouveau fichier
*I
        1:* PAUSE^G^G^GSONNE SONNE SONNE
        2:* ^C
*E
C>
```

Enfin, si vous créez le fichier de commandes avec un traitement de texte, reportez-vous à son manuel pour voir comment saisir une séquence de caractères ASCII. Nous verrons plus loin le jeu étendu de caractères ASCII (valeurs 128 à 255) qui permet à l'IBM PC et aux compatibles d'afficher des cadres et des symboles mathématiques.

Commande de traitement par lots PAUSE

Fonction :

Suspend le traitement d'un fichier de commandes après avoir affiché un message facultatif à l'utilisateur. Dès que celui-ci appuie sur une touche quelconque, le traitement par lots continue.

Format :

PAUSE [message]

Remarques :

Quand votre fichier de commandes exécute PAUSE, DOS affiche le message facultatif situé sur la ligne de commande PAUSE, suivi du message :

```
Appuyez sur une touche pour continuer . . .
```

Si l'utilisateur appuie sur une touche quelconque, l'exécution continue à la commande suivante du fichier de traitement par lots. Si l'utilisateur ne souhaite pas poursuivre le traitement par lots, il pourra l'interrompre en appuyant sur Ctrl+C puis sur O pour répondre à la question *Arrêt du traitement par lots (O/N) ?*

La commande de traitement par lots ECHO OFF supprime l'affichage du message facultatif figurant sur la ligne de commande PAUSE. Si votre fichier de commandes spécifie ECHO OFF, la commande PAUSE affiche toujours le message *Appuyez sur une touche pour continuer* . . . et suspend le traitement en attendant que l'utilisateur appuie sur une touche.

Exemple :

Le fichier de commandes invite l'utilisateur à insérer dans le lecteur A la disquette contenant les informations «paye» et à appuyer sur une touche quelconque pour continuer :

(suite)

```
PAUSE Insérez disque PAYE dans lecteur A
PAYE
```

Quand vous exécutez ce fichier, DOS affiche :

```
C> PAUSE  Insérez disque PAYE dans lecteur A
Appuyez sur une touche pour continuer . . .
```

AFFICHER DES MESSAGES PAR LA COMMANDE ECHO

La commande de traitement par lots PAUSE permet à un fichier de commandes d'afficher un message à l'utilisateur. Cependant, après chaque message que PAUSE affiche à l'écran, l'utilisateur doit appuyer sur une touche pour continuer. Dans de nombreux cas, vous préférerez des messages ou indicatifs qui évitent à l'opérateur une présence continue pour appuyer de temps en temps sur la touche Entrée. Une solution à ce problème est apportée par la commande de traitement par lots ECHO. Cette commande, nous l'avons vu, permet d'activer ou désactiver l'affichage des noms de commandes. Mais en outre, elle donne aux fichiers de commandes le moyen d'afficher une ligne de message à l'utilisateur. Si vous voulez utiliser ECHO pour afficher des messages, supprimez d'abord par ECHO OFF l'affichage des noms de commandes. Par exemple, un fichier de commandes appelé MESSAGE.BAT affiche les messages suivants :

```
Premier message
Deuxième message
Troisième message
```

Ce fichier se compose des commandes ci-dessous :

```
ECHO OFF
CLS
ECHO Premier message
ECHO Deuxième message
ECHO Troisième message
```

Le fichier de commandes se sert en premier lieu d'ECHO OFF pour supprimer l'affichage des noms de commandes. Si vous enlevez la ligne ECHO OFF, la sortie du fichier de commandes est inutilement encombrée :

```
C> ECHO Premier message
Premier message

C> ECHO Deuxième message
Deuxième message

C> ECHO Troisième message
Troisième message
```

L'aptitude des commandes ECHO à visualiser des messages trouve de nombreuses applications, depuis l'affichage de menus de traitement par lots jusqu'à la redéfinition des touches du clavier, en passant par le choix des couleurs de l'écran (les deux derniers exemples faisant appel au pilote de périphérique ANSI.SYS).

Remarquez, par exemple, comment procède la commande de traitement par lots ECHO pour afficher une mention de copyright dans le fichier de commandes suivant :

```
ECHO OFF
REM Nom : TOUSTOC.BAT
REM Fonction : Exécute les commandes de
REM traitement du stock.
REM
REM Ecrit par K. Jamsa 01/06/90
REM

REM Afficher mention de copyright.
ECHO TOUSTOC.BAT (copyright 1989 KJ Logiciel)
ECHO Tous droits réservés.

REM Le programme CALCSTOC détermine
REM la situation actuelle du stock.
CALCSTOC

REM Le programme TRISTOC produit une
REM liste triée du stock actuel.
TRISTOC
```

```
REM Le programme IMPSTOC imprime les
REM états sur papier du stock actuel.
IMPSTOC

REM Le programme ACHASTOC établit les
REM commandes de réapprovisionnement.
ACHASTOC
```

Quand l'utilisateur exécute ce fichier de commandes, DOS affiche :

```
ECHO OFF
TOUSTOC.BAT (copyright 1989 KJ Logiciel)
Tous droits réservés.
```

Vous constatez qu'ECHO vous permet d'adresser à l'utilisateur des messages significatifs sans interrompre le traitement par lots.

Les utilisateurs rencontrent souvent des difficultés quand ils tentent d'afficher des lignes vides à partir d'un fichier de traitement par lots utilisant la commande ECHO. Par défaut, si vous exécutez ECHO sans paramètre sur la ligne de commande (ni OFF, ni ON, ni message), ECHO rend compte de son état actuel ECHO ON ou ECHO OFF. Par exemple, à partir de l'indicatif de DOS, appelez la commande ECHO :

```
C> ECHO
Commande ECHO activée

C>
```

Si vous ne spécifiez aucun paramètre sur la ligne de commande, ECHO affiche simplement son état du moment. Un fichier de commandes appelé MONTRECO.BAT fait appel à ECHO pour afficher son état tout au long du traitement :

```
@ ECHO Afficher l'état par défaut
@ ECHO
@ ECHO Mettre ECHO à l'état OFF
@ ECHO OFF
@ ECHO
```

Quand vous exécutez ce fichier de commandes, DOS produit l'affichage ci-après :

```
C> MONTRECO
Afficher l'état par défaut
Commande ECHO activée
Mettre ECHO à l'état OFF
Commande ECHO désactivée
```

Même si vous mettez plusieurs caractères d'espacement après la commande ECHO, elle affiche toujours son état actuel. Le «secret» de l'affichage d'une ligne vide à partir de la commande ECHO réside dans le caractère (du jeu étendu) ayant 255 pour valeur du code ASCII. Votre ordinateur associe des caractères et symboles en principe distincts non seulement aux valeurs du code ASCII allant de 0 à 127, mais encore aux valeurs de 128 à 255 (jeu étendu de caractères). Le jeu standard de caractères ASCII associe le code 32 à l'espace. Si la ligne de commande ECHO ne contient que des espaces, ECHO affiche son état. Il se trouve que la valeur 255 du jeu étendu correspond elle aussi à un caractère d'espacement. La commande ECHO, cependant, ne reconnaît pas ce caractère comme un espace ordinaire. Il en résulte que, si vous mettez le caractère de code ASCII 255 sur la ligne de commande ECHO, celle-ci affichera la ligne vide souhaitée.

Créez le fichier de traitement par lots LIGNEVID.BAT en copiant les commandes à partir du clavier :

```
C> COPY CON LIGNEVID.BAT
```

Le fichier de commandes choisit d'abord l'état ECHO OFF et efface l'écran :

```
ECHO OFF
CLS
ECHO Sauter une ligne
```

Pour sauter une ligne, tapez le mot *ECHO* suivi d'un espace. Maintenez la touche Alt enfoncée et tapez 255 (en utilisant le pavé numérique). Lorsque vous relâchez la touche Alt, vous voyez le curseur se déplacer d'une position de caractère vers la droite. A présent, votre ligne de commande ECHO contient le caractère de code 255 dans le jeu étendu ASCII.

Utilisez la même procédure pour compléter le fichier de commandes :

```
C> COPY CON LIGNEVID.BAT
ECHO OFF
CLS
ECHO Sauter une ligne
ECHO <Alt+255 ici>
ECHO Sauter deux lignes
ECHO <Alt+255 ici>
ECHO <Alt+255 ici>
ECHO Dernière ligne
^Z
      1 Fichier(s) copié(s)

C>
```

Quand vous exécutez ce fichier de commandes, DOS affiche les lignes suivantes :

```
Sauter une ligne

Sauter deux lignes


Dernière ligne
```

Si ECHO affiche son état actuel quand ce fichier de commandes s'exécute, c'est que vous n'avez pas entré correctement la combinaison de touches Alt+255 pour les trois commandes ECHO, qui ne montrent donc aucun caractère associé (rappelez-vous qu'il faut absolument utiliser le pavé numérique).

Pour créer un fichier de commandes affichant une ligne vide en utilisant Edlin, faites appel à la même technique. Après avoir saisi le fichier et entré la commande ECHO, maintenez la touche Alt enfoncée et tapez 255 (avec le pavé numérique obligatoirement) :

```
C> EDLIN LIGNEVID.BAT
Nouveau fichier
*I
```



```
        1:* ECHO <Alt+255 ici>
```

Commande ECHO message

Fonction :

Affiche une ligne de message destinée à l'opérateur.

Format :

ECHO message

Remarques :

La commande ECHO, qui active ou désactive l'affichage des noms de commandes, permet aussi d'afficher des messages d'une ligne destinés à l'opérateur. Le message signalera, par exemple, une anomalie (un fichier non trouvé) ; mais il pourra aussi contenir des séquences «escape» ANSI qui effacent l'écran et déterminent ses couleurs. Tout au long de ce manuel de référence rapide, vous utiliserez ECHO pour dessiner des menus, pour faire retentir la «sonnerie» de l'ordinateur et même pour redéfinir les touches du clavier.

Exemple :

Ce fichier de commandes fait appel à ECHO pour émettre un signal sonore et avertir l'opérateur de la fin du traitement :

```
ECHO OFF
CALCSTOC
TRISTOC
IMPSTOC
ACHASTOC
ECHO ^G^G^GTraitement du stock terminé
ECHO Etat du stock courant : à l'impression
```

Souvenez-vous : Ctrl+G produit l'affichage ^G.

A la fin de l'exécution de ce fichier, la «sonnerie» de l'ordinateur retentit trois fois et DOS affiche :

```
Traitement du stock terminé
Etat du stock courant : à l'impression
```

Remarque : tandis que nous développerons, dans toute la suite du livre, les capacités des fichiers de commandes, ECHO apparaîtra clairement comme l'une des plus utiles parmi les commandes de traitement par lots.

OÙ PLACER VOS FICHIERS DE COMMANDES SUR DISQUE

Plus vous créez de fichiers de commandes et plus ils risquent de se répandre un peu partout sur votre disque. Nombreux sont les utilisateurs qui placent initialement leurs fichiers de commandes dans le même répertoire que celui où résident les commandes DOS. Ce choix n'est pas mauvais car, après tout, ce répertoire est normalement défini dans la commande PATH.

Une meilleure solution consiste cependant à créer un répertoire unique réservé aux fichiers de commandes d'usage fréquent. Vous pouvez l'appeler par exemple FICHCOMM ou UTIL (pour utilitaires) :

```
C> MKDIR \FICHCOMM
```

Après avoir copié vos fichiers de commandes vers le répertoire \FICHCOMM, vous pouvez modifier la commande PATH pour y ajouter ce répertoire dans le chemin de recherche. Si vous procédez ainsi, gardez présent à l'esprit le fait que DOS peut avoir à examiner chaque fichier du répertoire quand il recherche votre fichier de commandes. Si le répertoire contient plusieurs fichiers de commandes rarement utilisés, DOS perdra du temps à examiner ces noms de fichiers. C'est la raison qui conduit à n'inclure dans le répertoire que les fichiers de commandes d'usage fréquent.

CHAPITRE 4

Paramètres des fichiers de commandes

ABORDER LES PARAMÈTRES DES FICHIERS DE COMMANDES

Les fichiers de commandes sont là essentiellement pour vous permettre d'économiser à la fois du temps et de la frappe. Regardons un fichier de traitement par lots appelé I.BAT, qui imprime une copie du fichier AUTOEXEC.BAT. Le fichier I.BAT, très court, contient seulement la commande ci-dessous :

```
PRINT \AUTOEXEC.BAT
```

Dans ce cas, chaque fois que vous voulez imprimer le contenu du fichier AUTOEXEC.BAT, il vous suffit d'appuyer sur I, puis sur Entrée. Bien qu'il vous fasse gagner un nombre considérable de touches à frapper, il n'est pas très fonctionnel, car il s'applique à l'impression d'un fichier spécifique. Un fichier de commandes doté d'une plus grande souplesse vous permettrait d'abréger une commande telle que PRINT, et serait utilisable pour l'impression de *n'importe quel* fichier.

Chaque fois que vous tapez une commande après l'indicatif DOS, la ligne sur laquelle s'inscrit votre saisie est la ligne de commande. Une ligne de commande DOS se compose souvent de deux parties, un nom de commande tel que

DISKCOPY et des paramètres de ligne de commande tels que les identificateurs des lecteurs de disques A: et B:



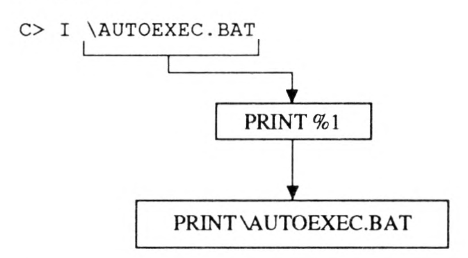
DOS met à votre disposition les moyens d'employer efficacement ces paramètres. Vous pourrez, par exemple, vous servir des paramètres pour créer rapidement le fichier de commandes I.BAT qui abrège la commande PRINT pour un fichier quelconque. Dans ce cas le fichier de commandes devient :

```
PRINT %1
```

Vous constatez que le fichier contient la commande PRINT de DOS, comme précédemment ; cependant, la commande donne à DOS l'ordre d'imprimer *%1* (au lieu d'un nom de fichier). Chaque fois que vous exécutez un fichier de commandes, DOS affecte le premier paramètre de la ligne de commande au paramètre rempaçable dont le symbole est %1. Dans ce cas, si vous exécutez I.BAT en tant que :

```
C> I \AUTOEXEC.BAT
```

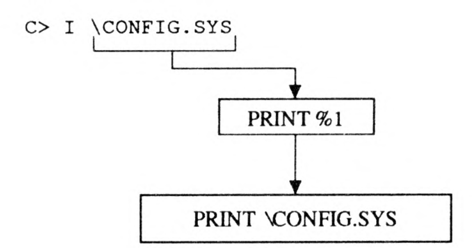
DOS affecte à %1 la valeur \AUTOEXEC.BAT, comme ci-dessous :



Si , après cela, vous exécutez ce fichier en tant que :

```
C> I \CONFIG.SYS
```

DOS affecte à %1 le nom de fichier \CONFIG.SYS, comme ceci :



Puisque la commande PRINT de DOS accepte les caractères génériques, vous pouvez exécuter le fichier en tant que :

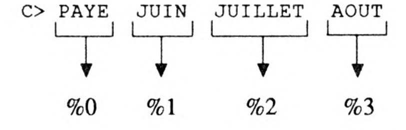
```
C> I *.BAT
```

Dans ce cas, DOS imprime tous les fichiers du répertoire courant dont les noms ont pour extension .BAT.

Nous venons d'accroître grandement la souplesse du fichier de traitement par lots en utilisant un paramètre rempaçable (ou *variable*). Pour faire passer les capacités de traitement à un échelon encore bien supérieur, DOS propose des paramètres allant de %0 à %9. Ainsi que vous venez de le voir, DOS affecte le premier paramètre d'une ligne de commande à la variable %1. Si votre ligne de commande contient plusieurs paramètres, comme ci-dessous :

```
C> PAYE JUIN JUILLET AOUT
```

DOS affecte les paramètres aux variables, en commençant à %1 et continuant jusqu'à %9 (en admettant qu'il y en ait autant). Chaque fois que vous exécutez un fichier de commandes, DOS affecte le nom de ce fichier à la variable %0. Dans notre exemple, les affectations deviennent :



Le fichier de commandes ci-dessous, que nous appellerons VOIRNOM.BAT, utilise la commande ECHO pour afficher son propre nom :

```
ECHO OFF
CLS
ECHO %0
```

Quand vous exécutez ce fichier, DOS affiche :

```
VOIRNOM
```

Grâce à la commande de traitement par lots ECHO, il est possible d'afficher les valeurs des paramètres %0 à %9 du fichier de commandes. Il est rare qu'un fichier de commandes ait besoin d'utiliser son propre nom au cours du traitement. Pour le cas où ce problème se présenterait, DOS met à votre disposition la variable %0.

Le fichier de commandes VOIRVAR.BAT ci-dessous utilise la commande ECHO pour afficher les paramètres correspondant aux variables %0 à %9 :

```
ECHO OFF
CLS
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

Si vous exécutez ce fichier avec la ligne de commandes :

```
C> VOIRVAR UN DEUX TROIS
```

DOS affiche :

```
C> VOIRVAR UN DEUX TROIS
VOIRVAR UN DEUX TROIS
```

De la même façon, si vous exécutez le fichier avec :

```
C> VOIRVAR A B C D E F G H
```

DOS affiche :

```
C> VOIRVAR A B C D E F G H
VOIRVAR A B C D E F G H
```

Les paramètres sont essentiels à l'élaboration de fichiers de commandes puissants. Le fichier de commande suivant,

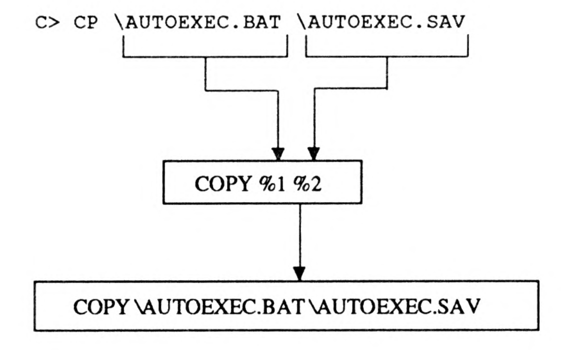
appelé CP.BAT, se sert des paramètres remplaçables %1 et %2 pour abréger la commande COPY :

```
COPY %1 %2
```

Si vous exécutez ce fichier de commandes en tant que :

```
C> CP \AUTOEXEC.BAT \AUTOEXEC.SAV
```

DOS affecte les paramètres comme ci-dessous :



Vous découvrirez utltérieurement l'emploi des commandes de traitement par lots IF et FOR. Vous pourrez constater alors à quel point les paramètres accroissent les qualités fonctionnelles des fichiers de commandes.

CHAPITRE 5

Programmation des fichiers de commandes

TESTER DES CONDITIONS SPÉCIFIQUES DANS UN FICHIER

Tous les fichiers de traitement par lots que nous avons examinés jusqu'ici commençaient leur exécution à la première commande, et continuaient de haut en bas jusqu'à la dernière. Lorsque la complexité des tâches à accomplir s'accroît, vous êtes amené, soit par un choix délibéré, soit par nécessité, à contrôler le déroulement de l'exécution de telle ou telle commande, en fonction d'un ensemble spécifique de conditions. La commande de traitement par lots IF donne à un fichier de commandes la capacité d'exécuter une commande DOS seulement quand une condition donnée est vraie.

La commande IF permet au fichier de commandes de tester six conditions distinctes : aux trois conditions IF proprement dites (EXIST, chaîne1==chaîne2 et ERRORLEVEL) viennent s'ajouter les trois autres conditions obtenues par l'opérateur NOT.

La première condition est l'existence d'un certain fichier sur le disque. La commande IF, pour déterminer si un fichier existe, est suivie du mot EXIST, du nom de fichier (avec son

extension) et de la commande à exécuter si la condition est vérifiée. La ligne suivante donne le format complet de la commande IF EXIST :

```
IF EXIST nomdefichier.ext CommandeDOS
```

Quand DOS rencontre la commande IF, il recherche le fichier spécifié sur le disque. Le paramètre *nomdefichier* contient éventuellement un chemin d'accès DOS complet avec la lettre d'identification du lecteur, suivie d'une cascade de sous-répertoires ; ou au contraire il s'agit d'un simple nom de fichier que DOS va rechercher dans le répertoire courant. Dans l'un et l'autre cas, si le fichier existe, DOS exécute la commande spécifiée. Si le fichier n'existe pas, DOS continue l'exécution à la commande suivante du fichier de traitement par lots.

La seconde condition IF vous permet de spécifier :

```
IF NOT EXIST nomdefichier.ext CommandeDOS
```

Lorsque la commande adopte cette forme, elle est exécutée si le fichier spécifié *n'existe pas*.

Si vous créez un fichier de commandes, nommé T.BAT, qui utilise le paramètre remplaçable %1 pour abréger la commande TYPE de DOS, comme ci-dessous :

```
TYPE %1
```

et si vous exécutez ensuite ce fichier de commandes avec un nom de fichier *confyg.sys* qui n'existe pas, DOS affiche le message :

```
Fichier introuvable - CONFYG.SYS
```

De la même façon, si vous exécutez le fichier de commandes sans spécifier le nom de fichier, DOS affiche le message :

```
Paramètre requis absent
```

Ce fichier de commandes se prête bien à l'utilisation de la commande de traitement par lots IF. Il peut commencer par

tester l'existence sur le disque du fichier spécifié. Dans l'affirmative, le fichier de commandes exécute la commande TYPE. Dans le cas contraire, DOS n'exécute pas la commande TYPE ; il n'y a pas de message d'erreur. Le nouveau fichier T.BAT contient :

```
IF EXIST %1 TYPE %1
```

Faites quelques essais sur ce fichier de commandes, avec d'une part des noms de fichiers existants et d'autre part des noms de fichiers absents.

Comme vous le savez, la commande COPY de DOS récrit un fichier existant sur le disque, sauf s'il s'agit d'un fichier pour lecture seulement, d'un fichier système ou d'un fichier caché. Pour éviter de récrire par inadvertance un fichier critique en manipulant la commande COPY, vous pourriez créer le fichier de commandes simple MACOPIE.BAT, qui se sert de la commande de traitement par lots IF EXIST pour déterminer si le fichier cible existe déjà. Si le fichier cible existe sur le disque, le fichier de commandes peut, à l'aide d'une commande PAUSE, avertir l'utilisateur qui appuiera sur une touche quelconque pour continuer l'opération de copie s'il le désire (note : l'opération de copie aura pour résultat le message *Accès refusé - NOMDEFICHIER.EXT* si le fichier possède un ou plusieurs des attributs «lecture seulement», «système», ou «caché» ; dans l'un quelconque de ces trois cas le fichier cible ne sera pas récrit). Si l'utilisateur ne souhaite pas poursuivre l'opération de copie, il peut appuyer sur Ctrl+C pour mettre fin à l'exécution du fichier de commandes. Si le fichier de commandes MACOPIE.BAT contient, par exemple :

```
IF EXIST %2 PAUSE Fichier cible existe déjà
COPY %1 %2
```

et si le fichier cible est présent sur le disque, le fichier de commandes affichera à l'écran la ligne de commande suivie d'un message à l'utilisateur (voir page suivante).

```
C> IF EXIST NOMDEFICHIER.EXT PAUSE Fichier
                                    cible existe déjà
Appuyez sur une touche pour continuer . . .
```

Note : par suite de contraintes de largeur, la ligne de commande C> ci-dessus déborde sur la ligne suivante.

L'opérateur peut annuler la commande en appuyant sur Ctrl+C

Vous pouvez même faire un pas de plus dans le traitement et inclure le paramètre remplaçable %2 dans le message affiché par PAUSE, comme ceci :

```
IF EXIST %2 PAUSE Fichier cible %2 existe déjà
COPY %1 %2
```

Dans ce cas, si l'utilisateur exécute le fichier de commandes avec AUTOEXEC.BAT comme fichier cible, le fichier de commandes affiche :

```
C> IF EXIST AUTOEXEC.BAT PAUSE Fichier cible
                              AUTOEXEC.BAT existe déjà
Appuyez sur une touche pour continuer . . .
```

A présent, l'utilisateur sait immédiatement pourquoi le fichier de commandes affiche le message PAUSE.

Note : par suite de contraintes de largeur, la ligne de commande C> ci-dessus déborde sur la ligne suivante.

Commande de traitement par lots IF EXIST

Fonction :

Teste l'existence d'un fichier et dans l'affirmative, exécute la commande DOS spécifiée.

Format :

IF EXIST nomdefichier.ext CommandeDOS

(suite)

Remarques :

La condition IF EXIST cherche sur le disque le fichier spécifié par un chemin complet (lecteur, répertoire, fichier) ou par un simple nom de fichier du répertoire courant.

S'il trouve le fichier, DOS exécute la commande spécifiée. Sinon, DOS continue l'exécution à la commande suivante.

Exemple :

Ce fichier de commandes vérifie si AUTOEXEC.BAT est présent dans le répertoire racine. Si oui, le fichier de commandes en imprime une copie et passe à la commande VER. Sinon, l'exécution continue à la commande VER :

```
IF EXIST \AUTOEXEC.BAT PRINT \AUTOEXEC.BAT
VER
```

La troisième condition IF teste l'égalité de deux chaînes de caractères (suites composées d'un ou plusieurs caractères). Le format de la commande est le suivant :

IF Chaîne1==Chaîne2 CommandeDOS

Si les chaînes de caractères de part et d'autre du double signe égal sont identiques, DOS exécute la commande spécifiée. Si les chaînes diffèrent, Dos continue l'exécution à la commande suivante du fichier de traitement par lots.

La quatrième condition IF teste *l'inégalité* de deux chaînes de caractères. Le format devient :

IF NOT Chaîne1==Chaîne2 CommandeDOS

Si DOS trouve deux chaînes différentes, la commande spécifiée est exécutée. Si les deux chaînes sont identiques, DOS continue l'exécution à la commande suivante du fichier.

Le fichier de commandes COULEUR.BAT, teste le nom de couleur spécifié par le premier paramètre remplaçable %1.

```
ECHO OFF
IF %1==ROUGE ECHO La carte est rouge
IF %1==BLEU ECHO La carte est bleue
IF %1==BLANC ECHO La carte est blanche
```

Exécutez le fichier en tant que :

```
C> COULEUR BLEU
```

et le fichier de commandes affichera :

```
La carte est bleue
```

Exécutez ensuite le fichier de commandes avec la couleur *ROUGE :*

```
C> COULEUR ROUGE
```

DOS détecte l'égalité des deux chaînes sur cette ligne :

```
IF %1==ROUGE ECHO La carte est rouge
```

Il en résulte que DOS affiche :

```
La carte est rouge
```

Pour que le fichier de commandes considère deux chaînes comme identiques, elles doivent correspondre caractère par caractère, et respecter notamment l'emploi des majuscules et minuscules. Si vous exécutez le fichier de commandes avec la couleur *Bleu* :

```
C> COULEUR Bleu
```

DOS ne trouve pas de correspondance entre couleurs, car la commande de traitement par lots IF ne considère pas les chaînes *Bleu* et *BLEU* comme identiques.

Un bon moyen de mieux comprendre le traitement effectué par la commande IF consiste à enlever ECHO OFF du fichier de commande précédent. Si vous exécutez alors le fichier de commandes en tant que :

```
C> COULEUR BLEU
```

vous pouvez voir les comparaisons réelles qu'effectue le fichier de commandes DOS.

Essayez ensuite d'exécuter le fichier de commande en omettant le paramètre de la ligne de commande :

```
C> COULEUR
```

A chaque tentative d'exécution d'une commande IF, DOS affiche le message :

```
Erreur de syntaxe
```

En effet, la syntaxe (ou format) de la commande IF n'est pas valide. Rappelez-vous : la commande IF compare deux chaînes. Si vous n'affectez aucune valeur au paramètre remplaçable %1, l'absence d'une des deux chaînes rend la comparaison impossible. DOS exigeant la présence de deux chaînes, l'absence de l'une d'elles provoque l'affichage d'un message d'erreur. Pour prévenir ce problème, vous devez faire en sorte que la commande ait toujours deux chaînes à comparer. Pour cela, placez chacune des chaînes entre *guillemets simples*, c'est-à-dire apostrophes :

```
IF '%1'=='ROUGE' ECHO La carte est rouge
IF '%1'=='BLEU' ECHO La carte est bleue
IF '%1'=='BLANC' ECHO La carte est blanche
```

Dans ce cas, si l'utilisateur entre la commande :

```
C> COULEUR BLEU
```

le fichier de commandes établit la correspondance :

```
IF 'BLEU'=='BLEU' ECHO La carte est bleue
```

Si l'utilisateur omet la couleur, la comparaison devient :

```
IF ''=='ROUGE' ECHO La carte est rouge
```

Les deux signes ' successifs (' ') correspondent à la «chaîne vide». La commande IF étant capable de comparer la chaîne vide à un nom de couleur, il n'y a pas d'erreur de syntaxe.

Le fichier de commandes TEST%1.BAT utilise la chaîne vide pour tester la valeur de %1. Si vous ne spécifiez aucun paramètre, le fichier de commandes affiche le message :

```
Valeur du paramètre absente
```

Dans beaucoup de fichiers de commandes, il sera nécessaire de vérifier si l'utilisateur a spécifié la valeur de %1 :

```
@IF '%1'=='' ECHO Valeur du paramètre absente
```

Si l'utilisateur exécute le fichier de commandes en tapant :

```
C> TEST%1
```

la condition IF devient :

```
IF ''=='' ECHO Valeur du paramètre absente
```

Commande de traitement par lots IF Chaîne1==Chaîne2

Fonction :

Vérifie si deux chaînes de caractères sont identiques et, dans l'affirmative, exécute la commande spécifiée.

Format :

IF Chaîne1==Chaîne2 CommandeDOS

Remarques :

La condition IF Chaîne1==Chaîne2 compare deux chaînes de caractères. Si tous les caractères coïncident, y compris les majuscules et minuscules, DOS exécute la commande placée immédiatement après la comparaison. Si un ou plusieurs caractères sont diffèrents, DOS continue le traitement à la ligne suivante du fichier de commandes.

Si vous ne spécifiez pas les deux chaînes sur la ligne de commande IF, DOS affiche le message d'erreur :

```
Erreur de syntaxe
```

Cette erreur se produit par exemple en testant un paramètre remplaçable dont la valeur est omise, comme ceci :

```
IF %1==SAUVEGARDE_MENSUELLE GOTO MENSUEL
```

(suite)

Dans ce cas, si l'utilisateur ne spécifie pas la valeur de %1, la condition devient :

```
IF ==SAUVEGARDE_MENSUELLE GOTO MENSUEL
```

Puisque la commande IF ne contient qu'une seule des deux chaînes, DOS affiche le message *Erreur de syntaxe*. La solution à ce problème consiste à mettre entre ***guillemets simples*** (c'est-à-dire entre apostrophes) les chaînes de caractères à comparer :

```
IF '%1'=='SAUVEGARDE_MENSUELLE' GOTO MENSUEL
```

Dans ce cas, si l'utilisateur ne spécifie aucune valeur pour le paramètre %1, DOS va comparer la chaîne vide à la chaîne SAUVEGARDE_MENSUELLE, comme le montre la ligne ci-dessous :

```
IF ''=='SAUVEGARDE MENSUELLE' GOTO MENSUEL
```

Puisque la ligne de commande contient deux chaînes, DOS ne produit pas de message d'erreur.

Exemple :

Le fichier de commandes ci-dessous détermine, à l'aide de la commande IF, la langue à utiliser :

```
IF '%1'=='ALL' GOTO Allemagne
IF '%1'=='ESP' GOTO Espagne
IF '%1'=='FRA' GOTO France
IF '%1'=='ITA' GOTO Italie
GOTO Pays_inconnu
```

Ce fichier de commandes compare la valeur du paramètre remplaçable %1 aux langues spécifiées (abrégées en trois lettres). Si la valeur de %1 correspond à l'une des langues, le fichier de commandes effectue un branchement à l'étiquette du pays correspondant. Si la valeur de %1 ne correspond à aucune des langues répertoriées, le branchement aboutit à l'étiquette Pays_inconnu.

La cinquième condition IF teste la sortie ou état terminal de la précédente commande DOS. Il se trouve que de nombreuses commandes DOS renvoient une valeur (ou code) de sortie qui indique si l'exécution s'est achevée avec succès et, sinon, pourquoi. Par exemple, la commande DISKCOPY de DOS renvoie l'une des valeurs de sortie suivantes :

Valeur	Etat terminal
0	Copie terminée avec succès
1	Copie a échoué, erreur non fatale disque
2	Copie incomplète, arrêt utilisateur par Ctrl+C
3	Copie a échoué, erreur fatale disque
4	Mémoire insuffisante ou lecteur non valide

La commande de traitement par lots IF ERRORLEVEL permet à vos fichiers de commandes d'examiner la valeur du code de sortie et de continuer le traitement en fonction de cette valeur. Le format de la commande IF ERRORLEVEL est le suivant :

```
IF ERRORLEVEL code CommandeDOS
```

Lorsqu'il rencontre la commande IF ERRORLEVEL, DOS examine la valeur du code de sortie de la commande précédente. Si la valeur du code de sortie est supérieure ou égale à la valeur spécifiée, DOS exécute la commande spécifiée. Si le code de sortie est inférieur à la valeur spécifiée, DOS continue le traitement à la commande suivante du fichier de traitement par lots. La sixième condition détermine si le code de sortie *n'est pas* égal ou supérieur à la valeur spécifiée. Si le code sortie est *inférieur* à la valeur spécifiée, la commande spécifiée est exécutée ; sinon, le traitement reprend à la commande suivante du fichier.

Un fichier de commandes, appelé NIVERR.BAT, vérifie si l'utilisateur a mis fin à la commande DISKCOPY par la combinaison Ctrl+C. Si c'est le cas, DISKCOPY renvoie la valeur 2 du code de sortie.

```
@DISKCOPY A: B:
@IF ERRORLEVEL 2 ECHO Arrêt par Ctrl+C
```

Exécutez le fichier de commandes NIVERR.BAT :

```
C> NIVERR
```

Quand DISKCOPY affiche le message :

```
Insérez la disquette source dans le lecteur A

Insérez la disquette cible dans le lecteur B

Appuyez sur une touche pour continuer . . .
```

appuyez sur Ctrl+C pour mettre fin à l'exécution de la commande. DOS affiche alors :

```
Arrêt par Ctrl+C
```

Souvenez-vous : pour mettre fin seulement à la commande en cours, en laissant le fichier de commandes poursuivre son exécution normalement, il faut appuyer sur N. Lorsque DOS continue l'exécution du fichier à la commande IF, la valeur du code de sortie de DISKCOPY est comparée à 2. Dans ce cas, puisque les valeurs sont égales, le fichier de commandes affiche le message :

```
Arrêt par Ctrl+C
```

Ne perdez pas de vue cet aspect : tant que la valeur du code de sortie de la commande précédente est supérieur ou égal à la valeur spécifiée, DOS exécute la commande associée à la commande IF. Notez que si DISKCOPY s'était arrêtée par suite soit d'*erreur fatale disque*, soit de *mémoire insuffisante ou lecteur non valide* (valeurs 3 et 4 du code), la condition aurait également été vérifiée, entraînant l'exécution de la commande DOS spécifiée.

C'est précisément leur capacité à tester et utiliser les valeurs du code de sortie qui nous permettra de bâtir les fichiers de commandes plus fonctionnels apparaissant plus loin dans ce livre. Votre but, quand vous créez un fichier de commandes, est d'automatiser une série de commandes DOS. Vos fichiers

de commandes devraient donc être capables de réagir en fonction d'erreurs imprévues. C'est exactement la faculté qu'IF ERRORLEVEL donne à vos fichiers de commandes.

Commande de traitement par lots IF ERRORLEVEL

Fonction :

Teste la valeur du code de sortie du programme précédent. Si cette valeur est supérieure ou égale à la valeur spécifiée, la commande DOS spécifiée est exécutée.

Remarques :

De nombreux programmes renvoient à DOS un code de sortie indiquant si l'exécution s'est achevée avec succès. Par exemple, la commande FORMAT de DOS renvoie les valeurs suivantes du code sortie :

Valeur	Signification
0	Formatage terminé avec succès
3	Formatage incomplet, arrêt utilisateur par Ctrl+C
4	Formatage incomplet, erreur
5	Arrêt utilisateur après le message suivant :

```
ATTENTION : TOUTES LES DONNEES DU DISQUE
DUR C SERONT PERDUES !
Lancement du formatage (O/N) ?
```

La commande de traitement par lots IF ERRORLEVEL permet à vos fichiers de commandes de tester la valeur du code de sortie d'un programme et de continuer ensuite le traitement en fonction du résultat.

Les commandes DOS ne fournissent pas toutes une valeur du code de sortie.

Quand il rencontre une commande IF ERRORLEVEL, DOS compare la valeur du code de sortie du programme précédent à la valeur spécifiée dans la commande IF. Si le code

(suite)

de sortie est supérieur ou égal à la valeur spécifiée, DOS exécute la commande correspondante. Si la valeur du code de sortie est inférieure à la valeur spécifiée, le traitement continue à partir de la commande suivante du fichier de traitement par lots (cette logique est inversée quand on utilise la sixième condition *IF NOT ERRORLEVEL*).

Exemple :

Ce fichier de commandes exécute la commande FORMAT de DOS et affiche un message donnant l'état final d'après le code de sortie de la commande FORMAT.

```
ECHO OFF
FORMAT A:
IF ERRORLEVEL 5 GOTO SANS_REPONSE
IF ERRORLEVEL 4 GOTO ERREUR
IF ERRORLEVEL 3 GOTO ARRET_CTRLC
ECHO Formatage terminé avec succès
GOTO FIN
:SANS_REPONSE
ECHO Le disque dur ne sera pas formaté
GOTO FIN
:ERREUR
ECHO Erreur traitement. Formatage incomplet
GOTO FIN
:ARRET_CTRLC
ECHO Formatage incomplet par arrêt Ctrl+C
:FIN
```

Note : la valeur la plus élevée du code est testée la première.

Important : si le code de sortie est supérieur ou égal à la valeur spécifiée, DOS exécute la commande spécifiée. Si ce fichier de commandes testait d'abord une valeur de 3, DOS effectuerait toujours le traitement correspondant à *arrêt Ctrl+C*, que le code ait pour valeur 3, 4 ou 5 (toujours supérieure ou égale à 3). En inversant l'ordre des tests, le fichier de commandes effectue le branchement à la position correcte pour chaque valeur du code de sortie.

UTILISER L'OPÉRATEUR *NOT*

Vous venez de le voir, la commande de traitement par lots IF permet à vos fichiers de commandes d'exécuter une commande DOS quand une condition spécifique est remplie. Cette condition est l'existence d'un certain fichier, l'identité de deux chaînes ou le code de sortie d'une commande supérieur ou égal à la valeur spécifiée. Dans de nombreuses applications des fichiers de commandes, il est plus commode pour DOS d'exécuter une certaine commande quand une condition testée *n'est pas* remplie. L'opérateur NOT des fichiers de traitement par lots vous donne le moyen de procéder ainsi.

L'opérateur NOT change le résultat d'une condition. Si, par exemple, une condition est vraie, l'opérateur NOT renvoie un résultat faux. De même, si le résultat d'une expression est faux, l'opérateur NOT renvoie une valeur vraie.

Considérez l'application de cette modification au fichier de commandes T.BAT. Ce fichier de traitement par lots vérifie d'abord si le fichier spécifié par le paramètre remplaçable %1 existe sur le disque. Dans l'affirmative, le contenu du fichier est visualisé. Le fichier de commandes effectue ensuite un deuxième test, qui permettra d'afficher le message suivant :

nomdefichier.ext n'existe pas

Le fichier de commandes T.BAT devient :

```
ECHO OFF
CLS
IF EXIST %1 TYPE %1
IF NOT EXIST %1 ECHO %1 n'existe pas
```

Dans ce cas, si le résultat de la condition EXIST %1 est faux (c'est-à-dire si le fichier n'existe pas), l'opérateur NOT produit un résultat vrai et donne l'ordre d'exécuter la commande ECHO. Si, au contraire, le résultat de la condition est vrai (le fichier existe), l'opérateur NOT donne un résultat faux. La commande ECHO n'est pas exécutée.

L'opérateur NOT est très largement utilisé dans les fichiers de commandes DOS. Par exemple, un fichier de commandes appelé COMPCH.BAT compare les deux chaînes contenues dans les paramètres remplaçables %1 et %2. Si elles sont identiques, le fichier de commandes affiche le message :

chaine1 et *chaîne2* identiques

Si les deux chaînes diffèrent, le fichier de commandes affiche le message :

chaine1 et *chaîne2* différentes

Les commandes de ce fichier sont les suivantes :

```
ECHO OFF
CLS
IF '%1'=='%2' ECHO %1 et %2 identiques
IF NOT '%1'=='%2' ECHO %1 et %2 différentes
```

Vous constatez que le fichier de commandes a besoin de deux chaînes pour effectuer son traitement. La première commande IF se charge de la condition où les deux chaînes sont les mêmes. La seconde commande IF traite la condition où les chaînes diffèrent (pour les programmeurs, ce processus est similaire à celui d'une instruction IF-ELSE).

Opérateur de traitement par lots NOT

Fonction :

Inverse le résultat vrai ou faux d'une commande de traitement par lots IF. Si le résultat est faux, NOT le rend vrai. De même, si le résultat est vrai, NOT le rend faux.

Format :

IF NOT condition CommandeDOS

Remarques :

L'opérateur NOT peut être utilisé avec l'une quelconque des trois conditions IF.

(suite)

```
IF NOT EXIST nomdefichier.ext CommandeDOS
IF NOT Chaîne1==Chaîne2 CommandeDOS
IF NOT ERRORLEVEL valeur CommandeDOS
```

Exemple :

Le fichier de commandes AX.BAT ci-dessous teste l'existence du fichier \AUTOEXEC.BAT. Si le test est négatif, le fichier de commandes affiche un message invitant l'utilisateur à créer le fichier \AUTOEXEC.BAT.

```
ECHO OFF
IF NOT EXIST \AUTOEXEC.BAT GOTO ABSENT
GOTO FIN
:ABSENT
ECHO Le répertoire racine ne contient pas le
ECHO fichier AUTOEXEC.BAT. Ce fichier de
ECHO commandes vous permet de spécifier une
ECHO ou plusieurs commandes que DOS doit
ECHO exécuter à chaque démarrage (le plus
ECHO souvent PRINT,PROMPT et PATH).
:FIN
```

Admettons que le fichier de commandes AUTOEXEC.BAT ne réside pas dans le répertoire racine ou sur le lecteur par défaut, notre fichier AX.BAT affiche :

```
Le répertoire racine ne contient pas le
fichier AUTOEXEC.BAT. Ce fichier de
commandes vous permet de spécifier une
ou plusieurs commandes que DOS doit
exécuter à chaque démarrage (le plus
souvent PRINT,PROMPT et PATH).
```

RÉPÉTER UNE COMMANDE DOS POUR UN JEU DE FICHIERS

Vous venez de le découvrir, la commande de traitement par lots IF permet à vos fichiers de commandes d'activer une

commande DOS quand une condition spécifique est remplie. Un traitement qui dépend du respect ou du non respect d'une telle condition est appelé *traitement conditionnel*. En plus du traitement conditionnel avec IF, DOS vous donne le moyen de répéter une certaine commande pour un certain jeu de fichiers. Un traitement qui s'exécute au moins une fois, avec possibilité de répétition, est appelé *traitement itératif*. Grâce à la commande FOR, vos fichiers de commandes répètent une commande pour un jeu spécifique de fichiers.

Le format de la commande de traitement par lots FOR est :

```
FOR %%Variable IN (Jeudefichiers) DO CommandeDOS
```

Comme toutes les commandes particulières au traitement par lots, FOR est une commande interne. DOS garde toujours en mémoire les commandes internes (par opposition aux commandes externes, comme par exemple DISKCOPY, qui résident sur disque).

La deuxième entrée de la ligne de commande FOR est la variable de traitement par lots %%Variable. Plus haut, nous avons vu les variables de traitement par lots (ou paramètres remplaçables) %0 à %9 auxquelles DOS substitue des valeurs au cours du traitement. %%Variable est similaire en cela que DOS affecte également une valeur à cette variable. La différence provient essentiellement de l'appellation : on utilise en général les lettres de l'alphabet pour identifier les variables de type %%Variable, sous la forme %%A ou %%F (une seule lettre autorisée). Le mot IN fait partie de la commande FOR. Il indique à DOS que les mots placés immédiatement après lui, entre parenthèses, définissent le jeu de fichiers à utiliser.

La commande de traitement par lots FOR fonctionne de la façon suivante : chaque fichier d'un jeu donné de fichiers est affecté successivement à %%Variable : une fois que DOS a affecté le premier nom de fichier, la commande DOS qui suit est exécutée. Quand l'exécution de cette commande est

terminée, DOS choisit dans le jeu de fichiers le nom de fichier suivant pour l'affecter à %%Variable et le processus se répète. Quand le jeu de fichiers est épuisé, l'exécution de la commande FOR est terminée.

Vous spécifiez le jeu de fichiers en tapant simplement les noms, séparés par des espaces ou des virgules. Voici quelques exemples de jeux valides pour la commande FOR :

(MAI.PAY JUIN.PAY JUILLET.PAY)

(MAI.PAY,JUIN.PAY,JUILLET.PAY)

(*.BAT *.EXE *.COM)

(*.*)

Vous voyez que FOR reconnaît les caractères génériques (ou jokers) de DOS, c'est-à-dire l'astérisque (*) utilisé dans l'exemple ci-dessus, aussi bien que le point d'interrogation (?). Quand FOR détecte la présence d'un joker, DOS le développe en un jeu de noms de fichiers. A son tour, FOR utilise ces noms l'un après l'autre.

Le mot DO signale à DOS que la commande à exécuter pour chaque fichier vient immédiatement après lui. Cette commande peut être n'importe quelle commande DOS. Pour mieux comprendre ce traitement, considérons la commande :

FOR %%A IN (A.BAT B.BAT C.BAT) DO TYPE %%A

Ici, FOR affecte d'abord le fichier A.BAT à la variable %%A et DOS exécute la commande TYPE %%A. Puisque le nom de fichier A.BAT a été affecté à la variable, DOS exécute en fait la commande TYPE A.BAT.

Une fois l'exécution de la commande TYPE terminée, FOR choisit le nom suivant dans le jeu de fichiers et l'affecte à la variable. Dans notre exemple, DOS affiche le contenu du fichier B.BAT. Quand l'exécution de TYPE est terminée, FOR répète à nouveau ce processus, en affectant le nom de fichier C.BAT à la variable.

Quand l'exécution de TYPE est terminée, FOR examine à nouveau le jeu de fichiers. Etant donné qu'il ne reste plus aucun fichier, DOS met fin à l'exécution de FOR.

De même, le fichier MONTRER.BAT ci-dessous se sert de la commande de traitement par lots FOR pour afficher tous les fichiers de commandes du répertoire courant :

```
FOR %%I IN (*.BAT) DO TYPE %%I
```

La commande FOR utilise la variable %%I. Le nom de variable dont vous vous servez importe peu. Nous l'avons déjà signalé, il ne doit cependant comporter qu'une unique lettre après le double symbole %%.

Par défaut, la commande DIR affiche, pour chaque fichier, les nom, extension, taille, date et heure de création :

```
COMMAND   COM    37637 17/06/88 12:00
COUNTRY   SYS    12838 17/06/88 12:00
DISKCOPY  COM    10428 17/06/88 12:00
DISPLAY   SYS    15741 17/06/88 12:00
FDISK     COM    70151 17/06/88 12:00
FORMAT    COM    22923 17/06/88 12:00
SORT      EXE     5914 17/06/88 12:00
   :       :       :       :        :
XCOPY     EXE    17087 17/06/88 12:00
```

Si vous appelez DIR suivi du paramètre /W, DIR n'affiche que les noms de fichiers et extensions, mais il présente cinq noms de fichiers dans la largeur de l'écran. Dans certains cas, vous préférerez afficher les noms de fichiers les uns au-dessous des autres comme ceci :

```
COMMAND.COM
COUNTRY.SYS
DISKCOPY.COM
DISPLAY.SYS
FDISK.COM
FORMAT.COM
SORT.EXE
   :
XCOPY.EXE
```

Le fichier de commandes DIRCOURT.BAT, ci-dessous, utilise la commande de traitement par lots FOR pour visualiser les noms de fichiers (grâce à ECHO) :

```
ECHO OFF
CLS
FOR %%I IN (*.*) DO ECHO %%I
```

Ici, la commande FOR affecte le nom de chaque fichier du répertoire courant à la variable %%I. DOS utilise ensuite la commande ECHO pour afficher le nom du fichier à l'écran. Nous l'avons vu, si votre commande contient des caractères jokers, DOS les développe en noms de fichiers complets.

Grâce aux paramètres remplaçables, nous pouvons améliorer la souplesse de DIRCOURT.BAT. Dans ce cas précis, au lieu de passer en revue tous les noms de fichiers du répertoire courant grâce aux astérisques (*.*), vous pouvez spécifier un nom de fichier comportant un ou deux astérisques comme premier paramètre de la ligne de commande. Par exemple :

```
C> DIRCOURT *.BAT
```

ou :

```
C> DIRCOURT *.TXT
```

Pour afficher tous les fichiers du répertoire courant, votre ligne de commande devient :

```
C> DIRCOURT *.*
```

Pour obtenir cette souplesse, vous modifiez simplement le fichier de commandes DIRCOURT.BAT comme ceci :

```
ECHO OFF
CLS
FOR %%I IN (%1) DO ECHO %%I
```

Le fichier de commandes TRI.BAT, destiné aux utilisateurs expérimentés de DOS, emploie les opérations de redirection pour afficher une liste triée des fichiers d'un répertoire, spécifiés par le premier paramètre de la ligne de commande.

```
ECHO OFF
IF EXIST TRIFICH.DON DEL TRIFICH.DON
FOR %%I IN (%1) DO ECHO %%I >> TRIFICH.DON
SORT < TRIFICH.DON
DEL TRIFICH.DON
```

Ce fichier de commandes utilise d'abord la condition IF EXIST pour détruire le fichier TRIFICH.DON s'il est présent sur le disque. Ensuite la commande FOR, grâce à ECHO, envoie vers l'écran chacun des noms de fichiers, comme le faisait le fichier de commandes précédent. Dans ce cas, cependant, le fichier de commandes fait appel à l'opérateur de redirection pour détourner de l'écran la sortie de la commande ECHO et venir l'ajouter à un fichier TRIFICH.DON. Une fois que FOR a créé ce «fichier des noms de fichiers», DOS utilise la commande SORT pour trier son contenu et inscrire à l'écran les noms de fichiers triés. Lorsque l'exécution de la commande SORT est terminée, le fichier TRIFICH.DON, devenu inutile, est détruit pour réinitialiser le fichier de commandes.

Avec la commande de traitements par lots FOR, vous pouvez améliorer le fichier de commandes T.BAT pour qu'il reconnaisse les caractères génériques de DOS :

```
ECHO OFF
CLS
FOR %%I IN (%1) DO TYPE %%I
```

Si vous procédez ainsi, la commande :

```
C> T *.BAT
```

affiche le contenu de tous les fichiers de commandes qui résident dans le répertoire courant. En fait, vous souhaiterez peut-être inclure plusieurs paramètres remplaçables dans le jeu de fichiers de la commande FOR, comme dans l'exemple ci-dessous :

```
ECHO OFF
CLS
FOR %%I IN (%1 %2 %3 %4 %5) DO TYPE %%I
```

La ligne de commande de votre fichier de traitement par lots peut maintenant spécifier plusieurs fichiers à visualiser :

```
C> T TRI.BAT DIRCOURT.BAT *.BAT
```

Le fichier de commandes affiche le contenu du fichier TRI.BAT, suivi du contenu de DIRCOURT.BAT. Puis il déploie les fichiers à caractère générique *.BAT et affiche tous les fichiers d'extension BAT situés dans le répertoire courant ou le lecteur courant.

La commande de traitement par lots FOR apporte une souplesse considérable aux fichiers de commandes. Nous l'utiliserons à plusieurs reprises dans la suite de cet ouvrage.

Commande de traitement par lots FOR

Fonction :

Répète une commande DOS pour un jeu donné de fichiers.

Format :

FOR %%V IN (Jeu) DO CommandeDOS

Remarques :

Quand vous exécutez la commande de traitement par lots FOR, celle-ci affecte à la variable spécifiée le premier fichier du jeu de fichiers spécifié. Le nom de la variable doit consister en une lettre unique. Vous spécifiez le jeu de fichiers en tapant les noms, séparés par des espaces ou des virgules. Après avoir affecté un nom de fichier à la variable, FOR active la commande DOS spécifiée. Quand l'exécution de cette commande est terminée, FOR affecte à la variable le nom suivant du jeu de fichiers et le processus se répète.

Quand le jeu de fichiers est épuisé, l'exécution de la commande FOR est terminée et le traitement continue à la ligne suivante du fichier de commandes.

(suite)

La commande FOR reconnaît totalement les cararactères génériques (ou «jokers») de DOS. Si le jeu de fichiers contient des jokers, la commande FOR éclate chacun d'eux en une liste appropriée de noms de fichiers qu'elle affecte, l'un après l'autre, à la variable.

Exemple :

Ce fichier de traitement par lots utilise la commande FOR pour activer la commande TYPE et afficher le contenu des fichiers A.TXT, B.TXT et C.TXT :

```
FOR %%I IN (A.TXT B.TXT C.TXT) DO TYPE %%I
```

De façon similaire, le fichier de commandes suivant utilise la commande TYPE pour afficher le contenu de tous les fichiers associés au paramètre %1. Si l'utilisateur exécute le fichier de commandes avec un caractère générique, comme l'astérisque dans *.BAT, le fichier de commandes affiche le contenu de chacun des fichiers correspondants :

```
FOR %%V IN (%1) DO TYPE %%V
```

EFFECTUER UN BRANCHEMENT À L'AIDE DE GOTO

Les fichiers de commandes que nous avons examinés tout au long de ce livre exécutaient toujours les commandes dans l'ordre d'écriture du fichier, en commençant à la première commande pour terminer à la dernière. Vous l'avez constaté en étudiant la commande de traitement par lots IF, on peut donner à DOS l'ordre d'exécuter une commande seulement quand une condition spécifique est remplie. Avec des applications plus complexes, vous pouvez avoir besoin occasionnellement d'un fichier de commandes qui exécute un certain jeu de commandes dans un cas et qui saute une ou plusieurs commandes dans un autre cas. La commande de

traitement par lots GOTO permet à votre fichier de commandes d'effectuer un branchement à partir d'une commande vers une autre commande. Le format de GOTO est :

GOTO Etiquette de traitement par lots

Observons le fonctionnement sur un exemple simple. Un fichier de commandes, nommé SUPPRIM.BAT, utilise le paramètre remplaçable %1 pour afficher le contenu d'un fichier. Il inscrit ensuite le message :

```
PAUSE nomdefichier.ext va être détruit
Appuyez sur une touche pour continuer . . .
```

Si l'utilisateur appuie sur une touche, le fichier est détruit. Si, au contraire, l'utilisateur ne veut pas détruire le fichier, il met fin à l'exécution du fichier de commandes en appuyant sur la combinaison de touches Ctrl+C.

```
ECHO OFF
IF '%1'=='' GOTO ABSENT
TYPE %1
PAUSE %1 va être détruit
DEL %1
GOTO FIN
:ABSENT
ECHO Spécifiez un fichier
:FIN
```

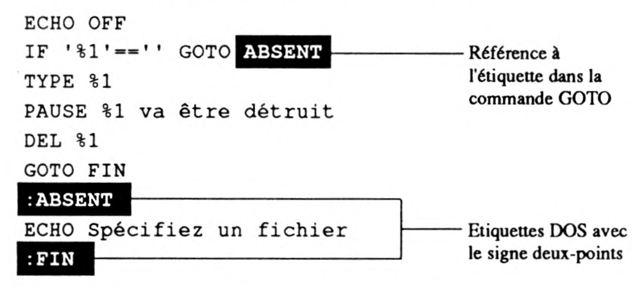
SUPPRIM.BAT, en premier lieu, vérifie si l'utilisateur a spécifié un fichier en tant que paramètre de la ligne de commande. S'il ne l'a pas fait, le fichier de commandes effectue un branchement vers l'étiquette :ABSENT, comme ceci :

```
ECHO OFF
IF '%1'=='' GOTO ABSENT
TYPE %1
PAUSE %1 va être détruit
DEL %1
GOTO FIN
:ABSENT
ECHO Spécifiez un fichier
:FIN
```

Dans ce cas, DOS affiche le message :

```
Spécifiez un fichier
```

et met fin à l'exécution. Vous voyez comment la commande de traitement par lots GOTO réalise un branchement pour transférer le contrôle à l'étiquette spécifiée. Une étiquette de traitement par lots commence par le signe deux-points et peut contenir un nombre quelconque de caractères, mais DOS ne reconnaît que les huit premiers. Ce fichier de commandes utilise les étiquettes :ABSENT et :FIN. Vous remarquerez que la ligne contenant l'étiquette commence par le signe deux-points, mais que ce signe est absent de la référence à l'étiquette, dans la commande GOTO :



Quand il rencontre, dans votre fichier de commandes, une ligne commençant par le signe deux-points, DOS identifie cette ligne comme étant une étiquette. Dans ce cas, DOS ne tente pas d'exécuter l'étiquette comme s'il s'agissait d'une commande. En outre, quel que soit l'état de la commande ECHO, les étiquettes ne sont pas affichées quand on exécute un fichier de commandes.

Avec SUPPRIM.BAT, si l'utilisateur appuie sur une touche pour détruire le fichier spécifié, le fichier de commandes émet la commande DEL et utilise GOTO pour se brancher à l'étiquette :FIN.

Vous le constaterez, les applications des fichiers de commandes sont pratiquement sans limites. Par exemple, le fichier de commande AIDEDOS.BAT constitue un utilitaire

d'aide en ligne portant sur DOS. Pour obtenir de l'aide sur une commande DOS particulière, l'utilisateur exécute AIDEDOS.BAT en spécifiant la commande. Par exemple :

```
C> AIDEDOS FORMAT
```

Le fichier de commandes examine le premier paramètre remplaçable (%1) et affiche le texte d'aide correspondant :

```
FORMAT commande

Type : externe

Fonction : prépare disque pour emploi par DOS

Format de la commande : FORMAT [lecteur :]
[/B][/1][/4][/8][/N:secteurs][/S][/T:pistes]
[/V[:nom]][/F:taille]

Exemple : FORMAT A:/4
```

Le fichier de commandes est très simple à écrire. Pour commencer, il vérifie si l'utilisateur a fourni une valeur pour le paramètre remplaçable. Si tel n'est pas le cas, la commande de traitement par lots GOTO effectue un branchement vers des commandes qui guident l'opérateur pour l'emploi du fichier AIDEDOS.BAT.

```
ECHO OFF
IF '%1'=='' GOTO INSTRUCTIONS
```

Supposons que l'utilisateur ait fourni une valeur pour le paramètre remplaçable. Le fichier de commandes teste ensuite chaque commande DOS pour déterminer celle pour laquelle une aide est demandée par l'utilisateur :

```
IF '%1'=='CLS' GOTO CLS_AIDE
IF '%1'=='FORMAT' GOTO FORMAT_AIDE
IF '%1'=='DISKCOPY' GOTO DISKCOPY_AIDE
:
IF '%1'=='XCOPY' GOTO XCOPY_AIDE
```

Vous constatez que, selon la valeur de %1, le fichier de commandes effectue un branchement vers une position

spécifique. Dans le cas de la commande CLS, par exemple, le fichier de commandes contient les lignes ci-après :

```
:CLS_AIDE
ECHO CLS commande
ECHO <Alt+255> (*)
ECHO Type : interne
ECHO <Alt+255> (*)
ECHO Fonction : efface l'écran,
ECHO amène le curseur en haut à gauche,
ECHO c'est-à-dire à la position origine
ECHO <Alt+255> (*)
ECHO Exemple : CLS
GOTO FIN
```

* Pour entrer le caractère ASCII 255, vous devez maintenir la touche Alt enfoncée et taper 255 sur le pavé numérique.

Si l'utilisateur exécute le fichier de commandes par :

```
C> AIDEDOS CLS
```

DOS produit l'affichage suivant :

```
CLS commande

Type : interne

Fonction : efface l'écran,
amène le curseur en haut à gauche,
c'est-à-dire à la position origine

Exemple : CLS
```

Après avoir affiché l'information d'aide sur un sujet spécifique, le fichier de traitement par lots utilise la commande GOTO pour se brancher à l'étiquette terminale :FIN.

Vous pouvez l'imaginer, un fichier de commandes complet, constituant un dispositif d'aide portant sur toutes les commandes DOS, atteindrait une taille respectable. Quant à elle, la version suivante du ficher AIDEDOS.BAT fournit une aide en ligne pour les commandes IF, FOR et GOTO :

```
ECHO OFF
IF '%1'=='' GOTO INSTRUCTIONS
IF '%1'=='IF' GOTO IF_AIDE
IF '%1'=='FOR' GOTO FOR_AIDE
IF '%1'=='GOTO' GOTO GOTO_AIDE
ECHO Commande incorrecte
:INSTRUCTIONS
ECHO Ce fichier de commandes offre une
ECHO aide en ligne pour les commandes
ECHO IF, FOR et GOTO. Pour l'exécuter,
ECHO tapez AIDEDOS nomdecommande, par
ECHO exemple AIDEDOS GOTO.
GOTO FIN
:IF_AIDE
ECHO IF commande
ECHO <Alt+255> (*)
ECHO Type : interne, traitement par lots
ECHO <Alt+255> (*)
ECHO Fonction : permet à un fichier de
ECHO            commandes de prendre des
ECHO            décisions grâce à l'une des
ECHO            conditions ci-dessous :
ECHO IF EXIST Nomdefichier.ext CommandeDOS
ECHO IF ERRORLEVEL CodeSortie CommandeDOS
ECHO IF Chaîne1==Chaîne2 CommandeDOS
ECHO <Alt+255> (*)
ECHO Exemple : IF EXIST %1 PRINT %1
GOTO FIN
:FOR_AIDE
ECHO FOR commande
ECHO <Alt+255> (*)
ECHO Type : interne, traitement par lots
ECHO <Alt+255>
ECHO Fonction : répète une commande DOS
ECHO            pour un jeu de fichiers
ECHO <Alt+255> (*)
ECHO Format : FOR %%V IN (jeu) DO CommandeDOS
ECHO <Alt+255> (*)
ECHO Exemple : FOR %%I IN (*.*) DO TYPE %%I
GOTO FIN
:GOTO_AIDE
ECHO GOTO commande
```

```
ECHO <Alt+255> (*)
ECHO Type : interne, traitement par lots
ECHO <Alt+255> (*)
ECHO Fonction : permet à un fichier de
ECHO commandes d'effectuer des branchements
ECHO d'une position à une autre.
ECHO <Alt+255> (*)
ECHO Format : GOTO étiquette
ECHO <Alt+255> (*)
ECHO Exemple : :REPETITION
ECHO            VER
ECHO            GOTO REPETITION
:FIN
```

* Pour entrer le caractère ASCII 255, vous devez maintenir la touche Alt enfoncée et taper 255 sur le pavé numérique.

CHAPITRE 6

Concepts avancés

UTILISER LES PARAMÈTRES NOMMÉS

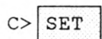
Ainsi que vous l'avez constaté, les paramètres remplaçables %0 à %9 de DOS accroissent les capacités de votre fichier de commandes, lui permettant de s'adresser à plusieurs applications. En plus de ces paramètres remplaçables, les versions 3.3 et ultérieures de DOS reconnaissent les *paramètres nommés*. Un paramètre nommé est une variable de traitement par lots placée entre deux signes %. Quand DOS rencontre un paramètre nommé dans votre fichier de commandes, il vérifie si une entrée correspondante figure dans la définition de son propre environnement. Par exemple, le fichier de commandes IMP.BAT imprime le fichier qui correspond au paramètre nommé %IMPFICH% :

```
PRINT %IMPFICH%
```

Quand DOS rencontre %IMPFICH% dans le fichier de commandes, il recherche dans la définition de ses paramètres d'environnement une entrée de la forme :

IMPFICH=

La commande SET de DOS vous permet de modifier ou d'afficher des entrées dans l'environnement DOS. Si vous émettez une commande SET non suivie d'une ligne de commande, comme ceci :



DOS affiche les entrées actuelles de l'environnement :

```
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
```

Pour créer une entrée d'environnement, vous utilisez SET, mais en affectant une valeur à l'entrée d'environnement :

```
C> SET IMPFICH=AUTOEXEC.BAT
```

Si vous exécutez à nouveau la commande SET (seule), DOS affiche sa nouvelle entrée :

```
C> SET
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
IMPFICH=AUTOEXEC.BAT
```

Si vous exécutez le fichier de commandes IMP.BAT, DOS remplace le paramètre nommé IMPFICH par l'entrée d'environnement AUTOEXEC.BAT et imprime le contenu du fichier correspondant.

Pour enlever une entrée d'environnement, utilisez la commande SET de la façon suivante :

```
C> SET IMPFICH=
```

Dans ce cas, DOS supprime l'entrée IMPFICH :

```
C> SET
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
```

Si vous exécutez à nouveau le fichier de commandes, IMP.BAT, DOS ne trouvera pas d'entrée correspondant à IMPFICH ; le paramètre nommé contiendra donc une chaîne vide. A l'aide de la commande de traitement par lots IF, le fichier de commandes peut faire un test de chaîne vide et poursuivre le traitement en fonction du résultat de ce test :

```
ECHO OFF
IF '%IMPFICH%'=='' GOTO PARAMABS
PRINT %IMPFICH%
GOTO FIN
:PARAMABS
ECHO Le paramètre nommé IMPFICH n'est pas
ECHO défini. Avec la commande SET de DOS
ECHO affectez-lui le fichier à imprimer.
:FIN
```

De nombreux utilisateurs ayant quelque difficulté à imaginer des applications des paramètres nommés, regardons un ou deux exemples. Considérons tout d'abord un fichier de commandes dont l'exécution créerait des fichiers temporaires. Certains utilisateurs souhaiteraient que DOS ne laisse pas ces fichiers dans le répertoire courant. Le fichier de commandes suivant crée un fichier temporaire appelé TRIDIR.TMP :

```
DIR ¦ SORT > TRIDIR.TMP
PRINT TRIDIR.TMP
MORE < TRIDIR.TMP
DEL < TRIDIR.TMP
```

Si l'utilisateur lance ce fichier de commandes, puis le termine par la combinaison de touches Ctrl+C avant la fin de l'exécution de la commande DEL, le fichier TRIDIR.TMP reste dans le répertoire courant. L'utilisateur pourrait chercher à résoudre ce problème en créant un répertoire TEMP destiné au stockage des fichiers temporaires et en se référant à ce répertoire tout au long du fichier de commandes, comme ceci :

```
DIR ¦ SORT > \TEMP\TRIDIR.TMP
PRINT \TEMP\TRIDIR.TMP
MORE < \TEMP\TRIDIR.TMP
DEL < \TEMP\TRIDIR.TMP
```

L'inconvénient de cette méthode est qu'elle oblige tous les utilisateurs à créer un répertoire TEMP pour loger les fichiers temporaires.

Le fichier de commandes TMPNOMME.BAT, ci-dessous, utilise un paramètre nommé pour désigner le lieu de stockage des fichiers temporaires :

```
DIR ¦ SORT > %TEMP%TRIDIR.TMP
PRINT %TEMP%TRIDIR.TMP
MORE < %TEMP%TRIDIR.TMP
DEL < %TEMP%TRIDIR.TMP
```

Quand il rencontre le paramètre nommé %TEMP%, DOS vérifie si l'environnement contient l'entrée correspondante.

Si une telle entrée n'existe pas, DOS affecte la chaîne vide à %TEMP%. Dans ce cas, DOS crée le fichier TRIDIR.TMP dans le répertoire courant. Si, au contraire, l'utilisateur crée une entrée d'environnement, telle que :

```
C> SET TEMP=D:
```

DOS utilise le paramètre nommé pour créer le fichier D:TRIDIR.TMP. Ce fichier de commandes répond aux besoins de deux catégories d'utilisateurs. Ceux qui acceptent de voir DOS créer des fichiers temporaires dans le répertoire courant exécutent simplement le fichier de commandes sans émettre de commande SET pour définir TEMP. En revanche, ceux qui souhaitent voir créer le fichier dans un répertoire spécifique, comme par exemple un «disque» rapide en mémoire vive (disque virtuel ou *RAM disk*), utilisent simplement la commande SET pour établir la correspondance entre le paramètre nommé et le lecteur ou sous-répertoire approprié.

Nous l'avons vu plus haut, l'entrée PATH= définit la liste des sous-répertoires où DOS recherche les programmes exécutables ou commandes externes. La plupart des utilisateurs définissent PATH= avec la commande PATH placée dans le fichier AUTOEXEC.BAT. Quand vous installez divers logiciels de provenance extérieure, le processus d'installation inclut souvent dans la commande PATH le répertoire nouvellement créé. Pour cela, l'installation modifie le contenu du fichier de commandes AUTOEXEC.BAT.

Supposons, pour notre exemple, que le fichier AUTOEXEC.BAT contienne les commandes suivantes :

```
ECHO OFF
PRINT /D:LPT1 /Q:32
PATH C:\DOS
```

Supposons en outre que l'installation d'un logiciel tende à ajouter le répertoire C:\UTIL à la commande PATH. Au lieu de modifier l'entrée :

```
PATH C:\DOS
```

le programme d'installation peut ajouter la ligne :

```
SET PATH=%PATH%;C:\UTIL
```

à la fin du fichier de commmandes. Cette entrée tire parti du fait que la commande PATH crée une entrée d'environnement DOS. Une fois que l'entrée PATH existe, vous pouvez l'utiliser comme paramètre nommé.

Pour mieux comprendre comment fonctionne ce traitement, créez un fichier de commandes appelé PATHPERS.BAT, qui contient :

```
ECHO OFF
SET PATHPERS=C:\DOS
ECHO Ajout de C:\UTIL à %PATHPERS%
SET PATHPERS=%PATHPERS%;C:\UTIL
ECHO L'entrée complète est %PATHPERS%
```

Quand vous exécutez ce fichier de commandes, DOS affiche les lignes ci-après :

```
Ajout de C:\UTIL à C:\DOS
L'entrée complète est C:\DOS;C:\UTIL
```

Vous constatez que le fichier de commandes a effectivement créé l'entrée complète C:\DOS;C:\UTIL, conformément à votre intention.

Dans toute la suite de ce livre, nous aurons l'occasion d'examiner plusieurs fichiers de commandes faisant appel aux paramètres nommés de DOS.

EXÉCUTER UN FICHIER DE COMMANDES À PARTIR D'UN AUTRE

Toutes les commandes exécutées par les fichiers de traitement par lots que nous avons rencontrés jusqu'ici étaient des fichiers dotés d'une extension EXE ou COM. Si votre fichier de commandes doit exécuter un autre fichier de commandes, vous devez employer l'une des commandes COMMAND et CALL de DOS, selon votre version du système.

Pour comprendre pourquoi votre fichier de commandes ne peut en exécuter un deuxième en appelant simplement son nom, comme il le ferait pour une commande DOS, créons le fichier de commandes élémentaires VERVOL.BAT, qui contient :

```
VER
VOL
```

et créons un fichier de commandes appelé PRIMAIRE.BAT qui contient :

```
DATE
VERVOL
TIME
```

PRIMAIRE.BAT exécute le fichier de traitement par lots VERVOL.BAT en se référant au nom de ce fichier. Quand vous lancez le fichier de commandes PRIMAIRE.BAT, DOS exécute d'abord la commande DATE et affiche donc les informations suivantes :

```
C> PRIMAIRE
C> DATE
La date courante est : Sam 27/01/90
Donnez la nouvelle date (jj/mm/aa) :
```

Si vous appuyez sur Entrée (laissant la date inchangée), DOS exécute le second fichier de commandes VERVOL.BAT et affiche les lignes ci-après :

```
C> PRIMAIRE
C> DATE
La date courante est : Sam 27/01/90
Donnez la nouvelle date (jj/mm/aa) :

C> VERVOL

C> VER

MS-DOS Version 4.00

C> VOL
```

```
Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE

C>
```

Le fichier de commandes VERVOL.BAT a terminé son exécution, mais DOS n'a pas lancé TIME, dernière commande du fichier de traitement par lots PRIMAIRE.BAT. En effet, si vous n'utilisez pas COMMAND ou CALL pour lancer un autre fichier de traitement par lots, DOS cesse d'exécuter des commandes dès que ce fichier secondaire a terminé sa propre exécution et tente de retourner au fichier qui l'a appelé.

Si vous utilisez DOS version 3.3 ou ultérieure, la commande CALL permet à un fichier de traitement par lots d'exécuter correctement les commandes d'un deuxième fichier de traitement par lots, puis de revenir à la commande suivante du fichier primaire. Le format de la commande CALL de DOS est le suivant :

CALL Fichier_de_commandes [paramètres]

Les paramètres facultatifs sont affectés aux «variables» %1 à %9 de DOS. Si vous utilisez DOS version 3.3 ou au-delà, changez le fichier de commandes PRIMAIRE.BAT en :

```
DATE
CALL VERVOL
TIME
```

Quand vous lancez ce fichier de commandes, DOS exécute à nouveau la commande DATE. Comme précédemment, appuyez sur Entrée (sans modifier la date système). DOS exécute la commande CALL, qui à son tour exécute le fichier de commandes VERVOL.BAT.

Lorsque l'exécution de VERVOL.BAT est terminée, DOS continue le traitement des commandes contenues dans PRIMAIRE.BAT, c'est-à-dire exécute la commande TIME. Il en résulte que DOS affiche :

```
C> PRIMAIRE
C> DATE
La date courante est : Sam 27/01/90
Donnez la nouvelle date (jj/mm/aa) :

C> CALL VERVOL

C> VER

MS-DOS Version 4.00

C> VOL

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE

C> TIME
L'heure courante est : 11:56:17,64
Donnez la nouvelle heure :

C>
```

Vous voyez que la commande CALL permet à un fichier de commandes d'exécuter correctement un second fichier de commandes et de revenir au premier pour y exécuter les commandes restantes.

Commande de traitement par lots CALL

Fonction :

Permet à un fichier de commandes d'exécuter un autre fichier de commandes et de revenir au fichier initial.

Format :

CALL Fichier_de_commandes [paramètres]

Remarques :

Si vous lancez un fichier de commandes à partir d'un autre sans utiliser CALL ou COMMAND /C (traité au paragraphe

(suite)

qui suit cet encadré), DOS cessera tout traitement dès qu'un fichier de commandes aura terminé son exécution. Si, au milieu d'un premier fichier de commandes, vous en lancez un deuxième, les commandes placées après le transfert au second fichier ne seront jamais exécutées. Dès que l'exécution du second fichier est terminée, DOS met fin à tout traitement.

Si vous utilisez une version 3.3 (ou ultérieure) de DOS, la commande de traitement par lots CALL vous permet d'exécuter un fichier de commandes à partir d'un autre et de revenir au fichier initial. Mettez simplement sur la ligne de commande CALL le nom du second fichier, ainsi que les paramètres s'il y a lieu.

Exemple :

Le fichier de commandes suivant utilise la commande de traitement par lots CALL pour exécuter le deuxième fichier IMBRIC.BAT :

```
VER
CALL IMBRIC
VOL
```

Dans ce cas, IMBRIC.BAT contient :

```
DATE
TIME
```

Si vous supprimez la commande CALL du premier fichier de commandes, en ne laissant que :

```
VER
IMBRIC
VOL
```

DOS n'exécutera jamais la commande VOL. Lorsque l'exécution d'IMBRIC.BAT est terminée, DOS arrête le traitement par lots et revient au message d'attente (indicatif) du système.

Si vous disposez d'une version de DOS plus ancienne que 3.3, vous devez vous servir de COMMAND /C pour exécuter un deuxième fichier de commandes. COMMAND.COM est un fichier DOS qui contient le processeur de la ligne de commande. Ce processeur exécute aussi bien les commandes que vous entrez après l'indicatif de DOS que les commandes contenues dans un fichier de traitement par lots. Si votre version de DOS est antérieure à 3.3, modifiez le fichier de commandes PRIMAIRE.BAT pour qu'il se compose des commandes suivantes :

```
DATE
COMMAND /C VERVOL
TIME
```

Vérifiez que le fichier COMMAND.COM réside effectivement dans le répertoire courant ou dans le chemin de recherche des commandes (commande PATH du fichier AUTOEXEC.BAT).

```
C> DIR COMMAND.COM

Volume dans unité C est DOSTROISUN
Répertoire de C:\DOS

COMMAND  COM      23674 22/04/85  12:00
        1 Fichier(s)
   21104460 octets disponibles

C>
```

Quand vous lancez le fichier de commandes, DOS exécute la commande DATE. Une fois de plus, appuyez sur Entrée pour accepter la date actuelle du système. Le traitement par lots se poursuit, utilisant COMMAND /C pour exécuter le fichier de commandes VERVOL.BAT. Quand le traitement de VERVOL.BAT se termine, DOS passe à l'exécution de la commande TIME dans le fichier PRIMAIRE.BAT. DOS produit l'affichage suivant :

```
C> PRIMAIRE

C> DATE
```

```
Date du jour : Sam 24/11/90
Entrez la nouvelle date
(jj/mm/aa) :

C> COMMAND /C VERVOL

C> VER

Ordinateur personnel IBM DOS ver. 3.10

C> VOL

Volume dans unité C est DOSTROISUN

C> TIME
Il est maintenant : 18:28:11,48
Entrez la nouvelle heure :

C>
```

COMMAND /C donne à DOS l'ordre de charger en mémoire un deuxième processeur de commandes. Les caractères /C sont là pour préciser que le processeur de commandes ne restera en mémoire que pour exécuter la commande placée immédiatement après eux. Dans notre exemple, le processeur de commandes secondaire reste en mémoire le temps nécessaire à l'exécution des commandes du fichier VERVOL.BAT. Dès que le traitement de ce fichier de commandes est terminé, DOS retire le processeur de commandes secondaire de la mémoire et exécute les commandes du fichier initial PRIMAIRE.BAT.

Commande de traitement par lots COMMAND /C

Fonction :

Permet à un fichier de commandes d'exécuter un autre fichier de commandes et de revenir au fichier initial.

Format :

COMMAND /C Fichier_de_commandes [paramètres]

(suite)

Remarques :

Si vous lancez un fichier de commandes à partir d'un autre sans utiliser CALL ou COMMAND /C, DOS cessera tout traitement dès qu'un fichier de commandes aura terminé son exécution. Si, au milieu d'un premier fichier de commandes, vous en lancez un deuxième, les commandes placées après le transfert au second fichier ne seront pas exécutées. Dès que le second fichier est exécuté, DOS met fin à tout traitement.

Si vous utilisez une version de DOS antérieure à 3.3, COMMAND /C vous permet d'exécuter un fichier de commandes à partir d'un autre et de revenir au fichier initial. Mettez simplement sur la ligne de commande COMMAND /C le nom du second fichier, accompagné éventuellement de paramètres.

Exemple :

Le fichier de commandes suivant utilise COMMAND /C pour exécuter le deuxième fichier IMBRIC.BAT :

```
VER
COMMAND /C IMBRIC
VOL
```

Dans ce cas, IMBRIC.BAT contient :

```
DATE
TIME
```

Si vous supprimez COMMAND /C du premier fichier de commandes, en ne laissant que :

```
VER
IMBRIC
VOL
```

DOS n'exécutera pas la commande VOL. Quand l'exécution d'IMBRIC.BAT est terminée, DOS arrête le traitement par lots et revient au message d'attente (indicatif) du système.

Vos fichiers de commandes vont devenir progressivement plus complexes. A l'occasion, vous choisirez peut-être de fournir à l'utilisateur un menu comportant plusieurs options. Chaque option de menu entraîne l'exécution d'une commande DOS différente. L'une d'entre elles permettra par exemple de suspendre l'exécution du fichier de traitement par lots pour émettre des commandes directement à partir de l'indicatif DOS. La commande COMMAND de DOS donne précisément cette possibilité à vos fichiers de traitement par lots.

Considérez un fichier de commandes INDICDOS.BAT, composé des commandes ci-dessous :

```
ECHO OFF
CLS
ECHO Vous êtes dans le fichier de commandes,
ECHO sur le point d'accéder à l'indicatif DOS
COMMAND
ECHO Retour au fichier de commandes
```

Quand DOS rencontre COMMAND, un processeur de commandes secondaire est chargé en mémoire et affiche un indicatif DOS invitant l'utilisateur à exécuter des commandes. Quand l'utilisateur n'a plus besoin d'entrer directement des commandes DOS, il lui suffit d'appeler la commande EXIT pour mettre fin au processeur de commandes secondaire.

Si le fichier COMMAND.COM est présent dans le répertoire courant ou dans le chemin de recherche des fichiers directement exécutables, exécutez le fichier INDICDOS.BAT. Dans ce cas, DOS affiche :

```
Vous êtes dans le fichier de commandes,
sur le point d'accéder à l'indicatif DOS

Microsoft(R) MS-DOS(R) Version 4.00
        (C)Copyright Microsoft Corp 1981-1988

C>
```

DOS a chargé un processeur de commandes secondaire, qui vous permet d'entrer des commandes après l'indicatif DOS. Dans cet exemple, choisissez la commande DATE pour afficher la date actuelle du système. Pour redonner le contrôle au fichier de commandes, utilisez la commande EXIT. Votre écran contient maintenant :

```
Vous êtes dans le fichier de commandes,
sur le point d'accéder à l'indicatif DOS

Microsoft(R) MS-DOS(R) Version 4.01
        (C)Copyright Microsoft Corp 1981-1988

C>DATE
La date courante est : Sam 24/11/90
Donnez la nouvelle date (jj/mm/aa) :
C> EXIT
Retour au fichier de commandes

C>
```

Quand l'exécution de la commmande EXIT est terminée, DOS redonne le contrôle au fichier de commandes, qui continue son traitement.

PLUS DE NEUF PARAMÈTRES AVEC LA COMMANDE *SHIFT*

Vous l'avez vu précédemment, les paramètres remplaçables %1 à %9 vous permettent de passer neuf valeurs à votre fichier de commandes. Lorsque vous utilisez cette possibilité, vos fichiers de commandes, plus souples, sont capables de couvrir des applications beaucoup plus nombreuses que les fichiers de commandes sans traitement de paramètres. Il arrive cependant, dans certains cas, que la restriction à neuf paramètres pose quelques problèmes. Pour de telles applications, la solution vous est apportée par la commande de traitement par lots SHIFT.

Le but principal de la commande SHIFT de DOS est de permettre aux fichiers de commandes d'accéder à plus de neuf paramètres remplaçables. Quand le fichier de commandes exécute SHIFT, DOS décale chaque paramètre remplaçable d'une position vers la droite. DOS affecte à %0 la valeur de %1, à %1 la valeur de %2, et ainsi de suite. Si la ligne de commande du fichier contient plus de neuf paramètres, SHIFT affecte la dixième valeur à %9. S'il ne reste aucune valeur, SHIFT affecte la chaîne vide à %9. Considérons, par exemple, un fichier de traitement par lots appelé DECAL1.BAT, qui contient :

```
ECHO OFF
CLS
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

Si vous exécutez ce fichier de commandes en tapant :

```
C> DECAL1 A B C
```

DOS affiche :

```
DECAL1 A B C
A B C
```

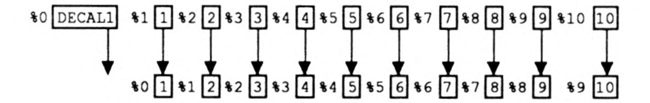
Si vous exécutez ce fichier de commandes avec plus de neuf paramètres, comme ci-dessous :

```
C> DECAL1 1 2 3 4 5 6 7 8 9 10 11
```

DOS affiche :

```
DECAL1 1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 10
```

Notez que SHIFT affecte la valeur 10 au paramètre %9. Graphiquement, l'opération SHIFT devient :



Le fichier de commandes suivant, DECAL2.BAT, utilise deux fois la commande SHIFT :

```
ECHO OFF
CLS
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

Si vous exécutez le fichier de commandes en tant que :

```
C> DECAL2 A B C
```

DOS affiche :

```
DECAL2 A B C
A B C
B C
```

Si, comme précédemment, vous exécutez le fichier de commandes avec plus de neuf paramètres, comme ceci :

```
C> DECAL2 1 2 3 4 5 6 7 8 9 10 11
```

DOS affiche :

```
DECAL2 1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 10
2 3 4 5 6 7 8 9 10 11
```

N'oubliez pas que, quand SHIFT n'a plus de valeur à affecter à %9, SHIFT affecte la chaîne vide à ce paramètre remplaçable. En faisant appel à un test de chaîne vide, vous pouvez créer un fichier de commandes appelé DECALIT.BAT qui, à l'aide de la commande GOTO, va «boucler» sur tous les paramètres remplaçables de la ligne de commande et afficher chacun d'eux sur sa propre ligne grâce à la commande ECHO :

```
ECHO OFF
:REPETE
ECHO %0
SHIFT
IF NOT '%0'=='' GOTO REPETE
```

Si vous exécutez ce fichier de commande en tant que :

```
C> DECALIT UN DEUX TROIS
```

DOS affiche :

```
DECALIT
UN
DEUX
TROIS
```

Quand l'exécution commence, %0 contient le nom du fichier de commandes. Quand la commande SHIFT est activée, DOS affecte le mot *UN* à %0. Comme %0 n'est pas égal à la chaîne vide, le fichier de commande active GOTO et ce processus se répète. Après la quatrième commande SHIFT, %0 contient la chaîne vide, ce qui met fin au traitement.

En utilisant la même technique, vous pouvez modifier légèrement le fichier DECALIT.BAT pour créer un fichier de commandes D.BAT, perfectionnement de la commande DIR :

```
ECHO OFF
CLS
:REPETE
DIR %1
SHIFT
IF NOT '%1'=='' GOTO REPETE
```

Vous pouvez maintenant exécuter le fichier de commandes en tant que :

```
C> D *.* *.BAT COMMAND.COM
```

Le fichier de commandes se charge d'afficher les listes de fichiers spécifiées. Si vous remplacez la ligne :

```
DIR %1
```

par :

```
FOR %%A IN (%1) DO TYPE %%A
```

la commande TYPE affiche le contenu de chaque fichier spécifié sur la ligne de commande du fichier D.BAT.

Commande de traitement par lots SHIFT

Fonction :

Décale chaque paramètre remplaçable d'une position vers la droite. Ainsi, DOS met dans %0 la valeur de %1. De même, DOS affecte à %1 la valeur de %2, à %2 la valeur de %3 et ainsi de suite.

Format :

SHIFT

Remarques :

Par défaut, DOS propose les paramètres remplaçables %0 à %9. Si la ligne de commande de votre fichier de traitement par lots contient plus de neuf paramètres, la commande SHIFT vous permet d'accéder à des paramètres supplémentaires. Supposons, par exemple, qu'un utilisateur exécute le fichier de commandes DECALTST.BAT, avec :

```
C> DECALTST A B C D
```

Le contenu des paramètres remplaçables %0 à %5 est :

%0	DECALTST
%1	A
%2	B
%3	C
%4	D
%5	' ' (chaîne vide)

Si le fichier de commandes émet une commande SHIFT, DOS décale chaque paramètre, ce qui donne :

%0	A
%1	B
%2	C
%3	D
%4	' ' (chaîne vide)
%5	' ' (chaîne vide)

(suite)

Si la ligne de commande contient plus de neuf paramètres, SHIFT place la première valeur non référencée dans %9 à chaque itération. Quand il ne reste plus de valeurs à affecter, SHIFT affecte la chaîne vide à %9.

Exemple :

Le fichier de commandes suivant emploie SHIFT pour afficher tous les paramètres que l'on passe sur la ligne de commande :

```
:REPETE
IF '%1'=='' GOTO EPUISE
ECHO %1
SHIFT
GOTO REPETE
:EPUISE
```

Le fichier de commandes parcourt simplement une boucle qui affiche la valeur de %1. Tant que %1 n'est pas la chaîne vide, la boucle se répète et une nouvelle valeur est affectée au paramètre remplaçable %1.

FICHIERS DE COMMANDES ET EXCEPTIONS

Vous connaissez toutes les commandes DOS de traitement par lots. Nous examinerons maintenant plusieurs problèmes que vous risquez de rencontrer lors de l'exécution de vos fichiers de commandes. Si un fichier de commandes affiche le message :

```
Nom de commande ou de fichier incorrect
```

c'est parce qu'il contient une commande non valide. Le moyen le plus rapide de déterminer la commande responsable du message d'erreur consiste à supprimer ECHO OFF, affichant ainsi les noms de commandes pendant l'exécution.

Le nom de commande incorrect apparaît à l'écran immédiatement avant le message d'erreur. Par exemple, le fichier de commandes MAUVCMDE.BAT contient une commande DOS non valide :

```
VER
XYZ12345
VOL
```

Quand vous lancez ce fichier de commandes, DOS affiche :

```
C> VER
MS-DOS Version 4.00
C> XYZ12345
Nom de commande ou de fichier incorrect
C> VOL
Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est OF35-18EE

C>
```

DOS affiche le message d'erreur immédiatement après la commande XYZ12345. Quand un fichier de traitement par lots contient une commande non valide, DOS affiche le message et continue l'exécution à la commande suivante. Dans bien des cas, la commande qui donne lieu à un message d'erreur est une commande DOS *valide*, mais DOS inscrit tout de même le message *Nom de commande ou de fichier incorrect.* Dans de tels cas, le problème n'est pas le nom de commande, mais plutôt la localisation par DOS du point de stockage de la commande sur le disque.

Rappelez vous : DOS définit des commandes internes d'une part, externes d'autre part. Les commandes internes (comme TYPE, DATE, CLS et VER) résident toujours en mémoire après le démarrage de DOS. Les commandes externes résident sur le disque. Les commandes d'usage courant DISKCOPY et FORMAT sont parmi les commandes externes. Pour exécuter l'une d'entre elles, DOS doit d'abord la trouver sur le disque, puis la charger en mémoire. Si la commande ne réside ni dans le répertoire courant, ni dans le

répertoire spécifié, DOS affiche le message *Nom de commande ou de fichier incorrect* uniquement parce qu'il ne parvient pas à localiser le fichier sur le disque. Ainsi que nous l'avons vu plus haut dans ce livre, la commande PATH augmente le nombre de lieux de stockage où DOS va rechercher des commandes. Si vous savez que la commande qui produit le message *Nom de commande ou de fichier incorrect est valide*, faites en sorte que DOS ait bien les informations nécessaires à la localisation de cette commande.

Finalement, il peut arriver que l'on rencontre le message *Nom de commande ou de fichier incorrect* même quand DOS parvient à localiser toutes les commandes figurant dans un fichier. Quand ce problème survient, il est vraisemblable que le traitement de texte utilisé a «noyé» dans le fichier de commandes un caractère étrange que DOS ne sait pas reconnaître. Si vous créez et sauvegardez vos fichiers de commandes avec un traitement de texte, vérifiez bien que le fichier est traité et sauvegardé en mode non document (ASCII).

Dans le cas où votre fichier de traitement par lots affiche le message ci-après :

```
Fichier introuvable
```

les commandes qu'il contient sont valides, mais DOS ne parvient pas à localiser un fichier que l'une des commandes tente d'appeler. Considérons le fichier de commandes MAUVFICH.BAT, qui contient :

```
TYPE 12345678.XYZ
```

Quand vous l'exécutez, DOS affiche :

```
Fichier introuvable

C>
```

Bien que DOS ait réussi à exécuter TYPE, cette commande n'a pas pu localiser le fichier 12345678.XYZ.

Plus les applications de vos fichiers de commandes vont devenir complexes et plus ils vont faire usage des paramètres remplaçables et des paramètres nommés. Grâce à la condition IF EXIST, vos fichiers de commandes pourront tester l'existence d'un fichier avant de tenter de s'en servir.

Vous vous souvenez sans doute que la commande de traitement par lots GOTO permet d'effectuer un branchement d'un point à un autre d'un fichier de commandes. Quand il rencontre la commande de traitement par lots GOTO, DOS commence par chercher, dans le fichier de commandes, l'étiquette spécifiée. Si DOS rencontre cette étiquette, il continue l'exécution du fichier de commandes à partir de la commande placée immédiatement après l'étiquette. Si, au contraire, l'étiquette n'existe pas, DOS met fin à l'exécution du fichier de commandes et affiche le message :

```
Etiquette introuvable
```

Si cette erreur se produit, vérifiez si votre fichier de commandes contient l'étiquette et si elle est orthographiée correctement. Le fichier de commandes ETIQABS.BAT montre que DOS arrête immédiatement le traitement quand une étiquette est non valide :

```
GOTO FIN
VER
VOL
```

Quand vous exécutez ce fichier de commandes, DOS affiche le message :

```
Etiquette introuvable
```

Vous remarquez que DOS n'a exécuté aucune des commandes VER et VOL.

Il est bon également de se souvenir que DOS n'utilise que les huit premières lettres d'un nom d'étiquette. Si DOS ne parvient pas à distinguer entre deux étiquettes, il choisira la première étiquette rencontrée. Un fichier de commandes

appelé 2ETIQ.BAT utilise les étiquettes *EtiquettUn et EtiquettDeux*. Les huit premières lettres de chaque étiquette étant les mêmes, DOS considère les étiquettes comme identiques :

```
:EtiquettUn
VER
GOTO EtiquettDeux
VOL
:EtiquettDeux
```

Quand vous exécutez ce fichier de commandes, DOS répète le numéro de la version courante de DOS, jusqu'à ce que vous utilisiez la combinaison de touches Ctrl+C pour terminer la commande. Bien que la commande GOTO se réfère à l'étiquette EtiquettDeux, DOS n'utilise que les huit premiers caractères de ce nom. Par suite, DOS répète indéfiniment le branchement au début du fichier de commandes.

TRAITEMENT PAR LOTS : COMMANDES PROPRES À OS/2

OS/2 exécute toutes les commandes de traitement par lots que nous avons examinées dans ce livre. Cependant, OS/2 offre trois commandes supplémentaires : EXTPROC, SETLOCAL et ENDLOCAL.

La commande OS/2 EXTPROC permet à votre fichier de traitement par lots de définir un processeur de commandes autre que l'interpréteur de la ligne de commande OS/2 pour exécuter les commandes du fichier. Le format de la commande EXTPROC est le suivant :

EXTPROC [lecteur:][chemin]nomdefichier.ext [arguments]

L'interpréteur de commandes qui sera utilisé est identifié par son nom complet (lecteur, cascade de répertoires, nom de

fichier, extension). Par exemple, ce fichier serait fourni par une société de logiciel. Les arguments facultatifs sont utilisés pour le démarrage de l'interpréteur de commandes. Ces paramètres dépendent entièrement du processeur de commandes dont vous vous servez.

EXTPROC, qui est là pour communiquer à OS/2 le nom de l'interpréteur de commandes, doit être la première commande du fichier de traitement par lots.

Commande de traitement par lots EXTPROC (OS/2)

Fonction :

Définit le processeur de ligne de commande qui sera utilisé par le fichier de traitement par lots pour émettre ses commandes.

Format :

EXTPROC [lecteur:][chemin]nomdefichier.ext [arguments]

Remarques :

La majorité des utilisateurs n'emploiera jamais EXTPROC. La commande de traitement par lots EXTPROC permet d'utiliser un logiciel fourni par une société extérieure pour exécuter les commandes d'un fichier de traitement par lots.

La commande EXTPROC définit le processeur de ligne de commande qui sera utilisé par le fichier de traitement par lots. Elle doit donc être la première commande de ce fichier.

Vous avez pu vous en rendre compte, les fichiers de commandes vous donnent le moyen de changer le lecteur et le répertoire par défaut, ainsi que les paramètres d'environnement de DOS. Dans certains cas, vous souhaiterez peut-être

que ces modifications n'aient lieu que pendant l'exécution du fichier de commandes. Par exemple, considérons un fichier de commandes, appelé MODIF.BAT, qui détermine la valeur FICHIER.EXT pour l'entrée d'environnement FILE, prend comme répertoire courant le répertoire racine du lecteur par défaut et finalement choisit A comme lecteur par défaut :

```
SET FILE=FICHIER.EXT
CHDIR \
A:
```

Toutes les modifications restent en vigueur lorsque l'exécution de ce fichier de commandes est terminée. Considérez, au contraire, une application qui émet ces trois commandes dans le cadre de son traitement global. Quand l'exécution du fichier de commandes est terminée, les lecteur et répertoire courants devraient être les mêmes que quand l'utilisateur a lancé le fichier de commandes. De même, l'environnement devrait avoir le même contenu qu'initialement.

Les commandes de traitement par lots d'OS/2 SETLOCAL et ENDLOCAL coopèrent pour sauvegarder et rétablir ultérieurement l'environnement d'OS/2, ainsi que les lecteur et répertoire courants.

Si vous utilisez la commande SETLOCAL, les modifications de l'environnement et des lecteur et répertoire courants durent seulement aussi longtemps que l'exécution du fichier de commandes. L'emploi de la commande ENDLOCAL permet, auparavant, de réinitialiser ces valeurs à l'intérieur du fichier de commandes.

Considérez un fichier de commandes NONMODIF.BAT, qui utilise SETLOCAL et ENDLOCAL :

```
SETLOCAL
SET FILE=NOUVNOM.EXT
CHDIR \
A:
ENDLOCAL
```

Quand il rencontre la commande de traitement par lots SETLOCAL, OS/2 sauvegarde le lecteur courant, le répertoire courant et le contenu de l'environnement. Quand il rencontre ensuite la commande de traitement par lots ENDLOCAL, OS/2 rétablit les réglages précédents. Il en résulte que les modifications propres au fichier de commandes sont limitées à la durée de vie de celui-ci ou restent en vigueur seulement jusqu'à la rencontre de la commande ENDLOCAL.

Commandes de traitement par lots SETLOCAL et ENDLOCAL (OS/2)

Fonction :

Sauvegardent et rétablissent ultérieurement le lecteur courant, le répertoire courant et le contenu de l'environnement OS/2.

Format :

```
SETLOCAL
  commandes de traitement par lots
ENDLOCAL
```

Remarques :

Au cours de leur traitement, de nombreux fichiers de commandes modifient le contenu de l'environnement, le lecteur courant ou le répertoire courant.

Dans la plupart des cas, ces modifications n'ont besoin d'être en vigueur que pendant la durée de traitement du fichier de commandes.

Malheureusement, les fichiers de commandes ne rétablissent que rarement les lecteur et répertoire courants en fin d'exécution. L'opérateur doit donc émettre d'autres commandes pour retrouver les lecteur et répertoire initiaux.

(suite)

Les commandes de traitement par lots SETLOCAL et ENDLOCAL (commandes propres à OS/2) se relaient pour sauvegarder et rétablir ultérieurement le lecteur courant, le répertoire courant et le contenu de l'environnement. La commande SETLOCAL sauvegarde les valeurs courantes. Une fois que l'exécution du fichier de commandes est terminée, la commande ENDLOCAL rétablit les valeurs initiales. Dans le cas où la commande SETLOCAL est utilisée et la commande ENDLOCAL omise, PS/2 rétablit les valeurs en fin d'exécution du fichier.

Exemple :

Le fichier de commandes suivant utilise SETLOCAL pour sauvegarder le lecteur courant, le répertoire courant et l'environnement. Le fichier de commandes modifie ensuite les lecteur et répertoire courants et émet diverses commandes SET pour modifier les entrées d'environnement. Avant la fin de son exécution, le fichier de traitement par lots active ENDLOCAL, qui rétablit les valeurs sauvegardées. Il en résulte que les modifications apportées au lecteur courant, au répertoire courant et au contenu de l'environnement sont seulement temporaires.

```
SETLOCAL
A:
CD \
SET FILE=TEST.BAT
SET PATH=A:
SET PROMPT=A$g
ENDLOCAL
```

Quand il rencontre la commande de traitement par lots SETLOCAL, OS/2 sauvegarde le lecteur courant, le répertoire courant et le contenu de l'environnement. Quand il rencontre ensuite la commande de traitement par lots ENDLOCAL, OS/2 rétablit les réglages précédents. Il en résulte que les modifications propres au fichier de commandes sont limitées à la durée de vie de celui-ci ou restent en vigueur seulement jusqu'à la rencontre de la commande ENDLOCAL.

Commandes de traitement par lots SETLOCAL et ENDLOCAL (OS/2)

Fonction :

Sauvegardent et rétablissent ultérieurement le lecteur courant, le répertoire courant et le contenu de l'environnement OS/2.

Format :

SETLOCAL
  commandes de traitement par lots
ENDLOCAL

Remarques :

Au cours de leur traitement, de nombreux fichiers de commandes modifient le contenu de l'environnement, le lecteur courant ou le répertoire courant.

Dans la plupart des cas, ces modifications n'ont besoin d'être en vigueur que pendant la durée de traitement du fichier de commandes.

Malheureusement, les fichiers de commandes ne rétablissent que rarement les lecteur et répertoire courants en fin d'exécution. L'opérateur doit donc émettre d'autres commandes pour retrouver les lecteur et répertoire initiaux.

(suite)

Les commandes de traitement par lots SETLOCAL et ENDLOCAL (commandes propres à OS/2) se relaient pour sauvegarder et rétablir ultérieurement le lecteur courant, le répertoire courant et le contenu de l'environnement. La commande SETLOCAL sauvegarde les valeurs courantes. Une fois que l'exécution du fichier de commandes est terminée, la commande ENDLOCAL rétablit les valeurs initiales. Dans le cas où la commande SETLOCAL est utilisée et la commande ENDLOCAL omise, PS/2 rétablit les valeurs en fin d'exécution du fichier.

Exemple :

Le fichier de commandes suivant utilise SETLOCAL pour sauvegarder le lecteur courant, le répertoire courant et l'environnement. Le fichier de commandes modifie ensuite les lecteur et répertoire courants et émet diverses commandes SET pour modifier les entrées d'environnement. Avant la fin de son exécution, le fichier de traitement par lots active ENDLOCAL, qui rétablit les valeurs sauvegardées. Il en résulte que les modifications apportées au lecteur courant, au répertoire courant et au contenu de l'environnement sont seulement temporaires.

```
SETLOCAL
A:
CD \
SET FILE=TEST.BAT
SET PATH=A:
SET PROMPT=A$g
ENDLOCAL
```

CHAPITRE 7

ANSI.SYS et DEBUG au service des fichiers de commandes

LIBÉRER LES APPLICATIONS DES FICHIERS DE COMMANDES À L'AIDE DU PILOTE DE PÉRIPHÉRIQUE *ANSI.SYS*

Si vous examinez vos disques DOS, vous y trouverez un fichier appelé ANSI.SYS :

```
C> DIR ANSI.SYS

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE
Répertoire de C:\DOS

ANSI      SYS       9148 17/06/88  12:00
        1 Fichier(s)  21231616 octets libres
```

L'extension SYS au nom de fichier vous indique que le fichier ANSI.SYS est un fichier du système d'exploitation. Plus précisément, ANSI.SYS est un pilote de périphérique qui exploite les fonctions étendues de l'écran et du clavier de l'ordinateur. Un *pilote de périphérique* est un programme, chargé en mémoire à chaque démarrage, qui permet au système de reconnaître et de gérer un périphérique associé à l'ordinateur. Une fois installé, le pilote offre des fonctions non accessibles par DOS lui-même.

Pour installer un pilote de périphérique, vous devez placer une entrée de type DEVICE= dans le fichier CONFIG.SYS. Vous vous rappelez que DOS utilise CONFIG.SYS à chaque démarrage du système, pour définir en mémoire sa propre configuration. Pour ANSI.SYS, votre entrée dans le fichier CONFIG.SYS devient :

```
DEVICE=ANSI.SYS
```

Vous devez fournir à DOS un chemin d'accès complet au fichier ANSI.SYS. Si donc ANSI.SYS réside dans un sous-répertoire DOS, votre entrée dans CONFIG.SYS devient :

```
DEVICE=C:\DOS\ANSI.SYS
```

N'oubliez pas qu'après une modification de CONFIG.SYS, cette modification ne prend effet que si vous relancez l'ordinateur.

Pour bénéficier, à partir d'un fichier de commandes, des capacités étendues qu'ANSI.SYS donne au clavier et à l'écran, vous devez employer soit la commande DOS ECHO, soit la commande DOS PROMPT pour envoyer des séquences d'échappement ANSI au pilote de périphérique.

Une *séquence d'échappement* est une combinaison unique de caractères qui commence par le caractère ASCII Escape (ASCII 27). Le pilote de périphérique ANSI.SYS reconnaît plusieurs séquences d'échappement pour régler les couleurs de l'écran, définir la position du curseur ou reprogrammer les touches du clavier.

Pour commencer, utilisons ANSI.SYS pour régler les couleurs de l'écran. La séquence d'échappement ANSI.SYS qui fixe la couleur de l'écran est :

Escape[couleur*m*

où *Escape* est le caractère ASCII Escape et *couleur* est l'un des numéros de couleur définis dans le tableau de la page suivante.

NUMEROS DE COULEURS RECONNUS PAR *ANSI.SYS*

Numéro	Couleur	
0	Couleur par défaut (noir et blanc)	
1	Attribut texte gras	
2 *	Attribut texte sous-brillance	
3 *	Attribut texte italique	
4	Soulignement pour IBM monochrome soulignement couleur (bleu) pour VGA	
5	Attribut texte clignotement	
6 *	Attribut texte clignotement rapide	
7	Attribut texte vidéo inverse	
8	Attribut texte caché	
30	Couleur de premier plan :	Noir
31		Rouge
32		Vert
33		Jaune
34		Bleu
35		Magenta
36		Cyan
37		Blanc
40	Couleur d'arrière-plan :	Noir
41		Rouge
42		Vert
43		Jaune
44		Bleu
45		Magenta
46		Cyan
47		Blanc
48 *	Indice	
49 *	Exposant	

* Non disponible sur VGA

Remarque : tous ces numéros ne sont pas reconnus par toutes les versions ANSI.SYS.

Par exemple, la séquence d'échappement ANSI.SYS :

```
Escape[31m
```

fixe le rouge comme couleur de premier plan de l'écran. De même, la séquence :

```
Escape[42m
```

choisit un fond vert. A l'aide de la commande PROMPT et du métacaractère $e, un fichier de commandes nommé APCYAN.BAT détermine un fond d'écran cyan :

```
PROMPT $e[46m
PROMPT [$p]
CLS
```

La commande PROMPT et son métacaractère $e permettent d'écrire le caractère ASCII Escape. La première commande détermine le cyan comme couleur d'écran, tandis que la seconde commande PROMPT ramène l'indicatif DOS aux lecteur et répertoire courants, tous deux affichés entre crochets. La commande CLS efface tout simplement l'écran et choisit cyan comme couleur d'arrière-plan.

De façon similaire, le fichier de commandes PPBLEU.BAT utilise la commande PROMPT pour choisir le bleu comme couleur de premier plan :

```
PROMPT $e[34m
PROMPT [$p]
CLS
```

Si vous exécutez ce fichier de commandes, le pilote de périphérique ANSI.SYS choisit le bleu comme couleur de premier plan, sans affecter la couleur d'arrière-plan.

Vous voyez que les fichiers de commandes APCYAN.BAT et PPBLEU.BAT sont presque identiques, la seule différence étant le numéro de couleur. On peut donc créer un seul fichier de commandes, appelé COULECR1.BAT, qui détermine les attributs de couleur écran en faisant appel au paramètre remplaçable %1. Dans ce cas, le fichier de commandes devient :

```
IF '%1'=='' GOTO FIN
PROMPT $e[%1m
PROMPT [$p]
CLS
:FIN
```

Pour déterminer une couleur d'écran, vous spécifiez simplement le numéro de couleur comme valeur d'un paramètre remplaçable. Par exemple, la ligne de commande :

```
C> COULECR1 42
```

choisit le vert comme couleur d'arrière-plan. DOS met simplement la valeur 42 dans le fichier de commande (et l'affecte à %1), ce qui donne la séquence d'échappement :

```
Escape[42m
```

Vous vous demandez peut-être dans quelles circonstances vos fichiers de commandes pourraient bien utiliser les attributs gras, clignotement ou vidéo inverse. Considérons une application qui affiche le message :

```
Le fichier FICHIER.EXT va être détruit
Appuyez sur une touche pour continuer . . .
```

Vous avez vu précédemment comment vous servir du signal sonore de l'ordinateur pour attirer l'attention de l'opérateur avant d'afficher un message par la commande PAUSE. Le fichier de commandes suivant, appelé DETRUIT.BAT, attire l'attention de l'opérateur par l'attribut texte clignotant :

```
ECHO OFF
IF '%1'=='' GOTO FIN
REM Choisir l'attribut texte clignotant.
REM Il faut mettre ECHO sur ON pour modifier
REM l'attribut texte par la commande PROMPT.
PROMPT $e[5m
ECHO ON
ECHO OFF
CLS
PROMPT [$p]
REM Affiche un message d'avertissement
ECHO %1 va être détruit
PAUSE
```

```
DEL %1
PROMPT  $e[0m
REM Rétablit la couleur par défaut. Il faut
REM à nouveu mettre ECHO sur ON.
ECHO ON
ECHO OFF
PROMPT [$p]
:FIN
```

Ce fichier de commandes affiche un message clignotant, conformément à ce qui a été spécifié, mais lorsque son exécution est terminée, il remet l'écran dans son état par défaut. Si vous avez utilisé le pilote de périphérique ANSI.SYS pour déterminer votre couleur d'écran, cette couleur est perdue.

Une solution à ce problème vous est apportée par le fichier de commandes COULECR2.BAT, qui peut définir les trois paramètres nommés ATTRTXT, PREPLAN et ARRPLAN de la façon suivante :

```
IF '%1'=='' GOTO FIN
PROMPT $e[%1m
PROMPT [$p]
CLS
ECHO OFF
IF '%1'=='0' SET ATTRTXT=0
IF '%1'=='1' SET ATTRTXT=1
IF '%1'=='2' SET ATTRTXT=2
IF '%1'=='3' SET ATTRTXT=3
IF '%1'=='4' SET ATTRTXT=4
IF '%1'=='5' SET ATTRTXT=5
IF '%1'=='6' SET ATTRTXT=6
IF '%1'=='7' SET ATTRTXT=7
IF '%1'=='8' SET ATTRTXT=8
IF '%1'=='30' SET PREPLAN=30
IF '%1'=='31' SET PREPLAN=31
IF '%1'=='32' SET PREPLAN=32
IF '%1'=='33' SET PREPLAN=33
IF '%1'=='34' SET PREPLAN=34
IF '%1'=='35' SET PREPLAN=35
IF '%1'=='36' SET PREPLAN=36
IF '%1'=='37' SET PREPLAN=37
IF '%1'=='40' SET ARRPLAN=40
```

```
IF '%1'=='41' SET ARRPLAN=41
IF '%1'=='42' SET ARRPLAN=42
IF '%1'=='43' SET ARRPLAN=43
IF '%1'=='44' SET ARRPLAN=44
IF '%1'=='45' SET ARRPLAN=45
IF '%1'=='46' SET ARRPLAN=46
IF '%1'=='47' SET ARRPLAN=47
IF '%1'=='48' SET ARRPLAN=48
IF '%1'=='49' SET ARRPLAN=49
:FIN
```

Vous constatez que, suivant le numéro de couleur spécifié comme premier paramètre de COULECR2.BAT, le fichier de commandes définit l'un des trois paramètres nommés.

Quand vous créez un autre fichier de commandes, tel que DETRUIT.BAT, qui modifie les attributs d'écran, ce fichier peut verifier par un test si les paramètres nommés existent. Dans l'affirmative, il peut rétablir les couleurs d'écran initiales.

```
ECHO OFF
IF '%1'=='' GOTO FIN
REM Choisir l'attribut texte clignotant.
REM Il faut mettre ECHO sur ON pour modifier
REM l'attribut texte par la commande PROMPT.
PROMPT $e[5m
ECHO ON
ECHO OFF
CLS
PROMPT [$p]
REM Affiche un message d'avertissement
ECHO %1 va être détruit
PAUSE
DEL %1
REM Rétablit la couleur par défaut. Il faut
REM à nouveau mettre ECHO sur ON.
ECHO ON
IF NOT '%ATTRTXT%'=='' prompt $e[%ATTRTXT%m
IF NOT '%PREPLAN%'=='' prompt $e[%PREPLAN%m
IF NOT '%ARRPLAN%'=='' prompt $e[%ARRPLAN%m
ECHO OFF
CLS
PROMPT [$p]
:FIN
```

Nous l'avons vu, la commande ECHO permet elle aussi d'envoyer des séquences d'échappement au pilote de périphérique ANSI.SYS. Pour créer un fichier de commandes écrivant des séquences d'échappement avec ECHO, le plus simple est d'employer Edlin. C'est ainsi que vous pouvez par exemple écrire le fichier de commandes EFFECRAN.BAT, qui fait appel à la séquence d'échappement ANSI.SYS :

Escape[2J

pour effacer l'affichage écran et ramener le curseur en haut à gauche (à la position origine). Comme il a été indiqué plus haut, lancez Edlin avec le nom de fichier, comme ceci :

```
C> EDLIN EFFECRAN.BAT
```

Après l'indicatif * d'Edlin, tapez I, commande d'insertion d'Edlin, et appuyez sur Entrée. Après les caractères 1:* tapez le mot ECHO et appuyez une fois sur la barre d'espacement. DOS affiche :

```
C> EDLIN EFFECRAN.BAT
Nouveau fichier
*I
     1:* ECHO
```

Ensuite, il faut entrer la séquence d'échappement ANSI.SYS qui efface l'écran. Pour entrer le caractère ASCII Escape, maintenez la touche Ctrl enfoncée et appuyez sur V :

```
C> EDLIN EFFECRAN.BAT
Nouveau fichier
*I
     1:* ECHO ^V
```

Relâchez Ctrl, tapez deux fois [et enfin tapez *2J* :

```
C> EDLIN EFFECRAN.BAT
Nouveau fichier
*I
     1:* ECHO ^V[[2J
```

Edlin utilise les caractères ^V[pour écrire le caractère ASCII Escape. Appuyez sur Entrée, puis Ctrl+C pour sortir

du mode insertion d'Edlin. Sauvegardez le fichier de commandes en choisissant E, commande Fin d'Edlin. Quand vous lancez EFFECRAN, la commande ECHO écrit la séquence d'échappement qui efface votre écran (bien entendu, il est encore plus facile de choisir la commande CLS de DOS. Le but d'EFFECRAN.BAT était de vous montrer comment créer une séquence d'échappement par la commande ECHO).

Le pilote de périphérique ANSI.SYS qui, nous venons de le voir, efface l'écran ou détermine ses couleurs de premier plan et d'arrière-plan, permet aussi à vos fichiers de commandes de contrôler la position du curseur sur l'écran. Une fois qu'ANSI.SYS a défini la position du curseur, la sortie suivante vers l'écran se produira à cette position.

Le tableau ci-dessous définit les cinq routines ANSI.SYS de déplacement du curseur :

Séquence d'échappement	Résultat
Escape[NombreLignesA	Curseur vers le haut
Escape[NombreLignesB	Curseur vers le bas
Escape[NombreColonnesC	Curseur vers la droite
Escape[NombreColonnesD	Curseur vers la gauche
Escape[Ligne;ColonneH	Curseur à la position (Ligne, Colonne)

La plupart des écrans ont vingt-cinq lignes et quatre-vingt colonnes. La position extrême en haut à gauche de l'écran est 1,1 (c'est-à-dire ligne 1, colonne 1). La position extrême en bas à droite est 25, 80. Le fichier de commande suivant, appelé POSICUR.BAT, utilise les séquences d'échappement ANSI :

```
Escape[1;1H
Escape[5;5H
Escape[10;10H
Escape[20;20H
```

pour afficher des messages aux positions ligne, colonne : (1, 1), (5, 5), (10, 10) et (20, 20). Le fichier de commandes utilise la commande de traitement par lots ECHO pour définir la position du curseur :

```
ECHO OFF
CLS
ECHO Escape[1;1HLigne 1, Colonne 1
ECHO Escape[5;5HLigne 5, Colonne 5
ECHO Escape[10;10HLigne 10, Colonne 10
ECHO Escape[20;20HLigne 20, Colonne 20
```

Comme il n'est pas facile d'afficher le caractère ASCII Escape dans un texte, ce fichier emploie le mot *Escape* chaque fois que vous devez inclure le caractère ASCII Escape. Si vous utilisez Edlin pour créer vos fichiers de commandes, remplacez chaque *Escape* par la combinaison de caractères ^V[.

Nous avons créé précédemment un fichier de commandes AIDEDOS.BAT, qui affiche une aide en ligne pour chaque commande DOS. Le pilote de périphérique ANSI.SYS vous permet de modifier ce fichier de commandes pour présenter les noms de commandes en caractères gras centrés et le texte d'aide proprement dit dans une couleur de premier plan différente.

Ce fragment de fichier de commandes illustre l'utilisation du pilote de périphérique pour l'aide sur la commande CLS :

```
ECHO OFF
CLS
REM Supprimer tous les attributs.
ECHO Escape[0m
REM Sélectionner le texte en gras.
ECHO Escape[1m
REM Centrer le mot CLS.
ECHO Escape[1;35HCLS
REM Choisir le bleu comme couleur de texte.
ECHO Escape[34m
ECHO Escape[3;10HType : Interne
```

```
ECHO Escape[5;10HFonction : Efface l'écran et
ECHO place le
ECHO Escape[6;20Hcurseur en haut à gauche, à
ECHO la position origine.
ECHO Escape[8;10HExemple : CLS
```

Dans cet exemple, le fragment de fichier utilise le pilote de périphérique ANSI.SYS pour définir l'attribut texte gras et ensuite pour centrer le nom de commande CLS. Le fichier de commandes inhibe tous les attributs actuellement en vigueur pour éviter les conflits potentiels de couleurs. Il choisit la couleur de premier plan et affiche le texte d'aide. Vous voyez que le fichier de commande utilise la séquence d'échappement :

Escape[Ligne;ColonneH

pour afficher chaque ligne d'aide en commençant à la colonne 10 (les fins de lignes commencent à la colonne 20). Une fois le texte affiché, le fichier de commandes pourrait utiliser le pilote ANSI.SYS pour réinitialiser les attributs vidéo (vous verrez ultérieurement comment créer et utiliser des menus depuis vos fichiers de commandes ; vous ferez alors très largement appel aux séquences d'échappement pour déterminer la position du curseur).

Le pilote de périphérique ANSI.SYS développe les capacités de l'écran et du clavier à la fois. Vous venez de l'utiliser pour effacer l'écran, déterminer les couleurs de premier plan et d'arrière-plan, définir la position du curseur. Pour ouvrir l'éventail des fonctions du clavier, ANSI.SYS vous permet de redéfinir les touches. La plupart des utilisateurs assignent des chaînes de caractères aux touches de fonction F7 à F10, qui ne sont pas prédéfinies par DOS. Souvenez-vous : DOS utilise les touches de fonction F1 à F6 pour l'édition de la ligne de commande.

A chaque touche de votre clavier est associée une valeur numérique distincte ; cette valeur est appelée *code clavier*.

Les touches F7 à F10 ont pour valeurs du code 65, 66, 67 et 68 respectivement. Pour assigner une chaîne de caractères à l'une de ces touches de fonction, votre séquence d'échappement ANSI.SYS devient :

```
Escape[0;Codeclavier;"chaîne"p
```

Par exemple, pour assigner la commande DOS VER à la touche de fonction F7, votre ligne de commande devient :

```
Escape[0;65;"VER"p
```

De même, pour assigner la commande CLS à la touche de fonction F10, la séquence d'échappement devient :

```
Escape[0;68;"CLS"p
```

Le fichier de commandes DEFTOUCH.BAT ci-dessous utilise la séquence d'échappement ANSI.SYS de réassignement des touches pour définir une touche de fonction :

```
IF '%1'=='' GOTO FIN
IF '%2'=='' GOTO FIN
ECHO Escape[0;%1;"%2"p
:FIN
```

Ce fichier de commandes utilise le paramètre remplaçable %1 pour déterminer la touche à définir et le paramètre %2 pour déterminer la valeur à assigner à la touche. Par exemple, pour assigner la commande CLS à la touche de fonction F10, votre ligne de commande devient :

```
C> DEFTOUCH 68 CLS
```

Vous constatez que le fichier de commandes substitue les valeurs *68* et *CLS* aux paramètres remplaçables dans la séquence d'échappement :

```
ECHO Escape[0;68;"CLS"p
```

Dans bien des cas, vous souhaiterez assigner plusieurs mots à une touche. Par exemple :

```
C> DEFTOUCH 68 DIR *.* /P
```

Pour cela, modifiez simplement le fichier de commandes DEFTOUCH.BAT pour utiliser les paramètres %1 à %9, comme ci-dessous :

```
IF '%1'=='' GOTO FIN
IF '%2'=='' GOTO FIN
ECHO Escape[0;%1;"%2 %3 %4 %5 %6 %7 %8 %9"p
:FIN
```

En assignant des commandes d'usage fréquent aux touches de fonction F7 à F10, vous gagnez du temps et de la frappe.

Rappelez-vous : après avoir créé le fichier de commandes DEFTOUCH.BAT, vous pouvez le lancer à partir du fichier AUTOEXEC.BAT en l'appelant soit par COMMAND /C, soit par CALL. En procédant ainsi, vous activez vos définitions de touches à chaque démarrage du système.

Vous avez maintenant un aperçu de la réaffectation des touches et des moyens de l'accomplir. Soyez attentif au fait que le premier paramètre (zéro), dans ces exemples, est exigé quand le code clavier est une valeur du code ASCII étendu. Pour les valeurs qui *ne correspondent pas* à un code ASCII étendu, on ne doit pas employer le zéro et son point-virgule.

Les codes ASCII étendus sont définis ci-après :

15	Maj+Tab
16-25	Alt+q, w, e, r, t, y, u, i, o, p
30-38	Alt+a, s, d, f, g, h, j, k, l
44-50	Alt+z, x, c, v, b, n, m
59-68	F1-F10
71	Origine
72	Curseur vers le haut
73	Page précédente
75	Curseur vers la gauche
77	Curseur vers la droite
79	Fin

80	Curseur vers le bas
81	Page suivante
82	Insertion
83	Suppression
84-93	Maj+F1 à Maj+F10
94-103	Ctrl+F1 à Ctrl+F10
104-113	Alt+F1 à Alt+F10

EXPLOITER AVEC *DEBUG* LE POTENTIEL DES FICHIERS DE COMMANDES

Toutes les commandes DOS ont été écrites par des programmeurs à l'aide d'un langage tel que C ou Pascal. Parmi les trente millions d'utilisateurs de DOS, bien peu sont programmeurs. Cependant, pour parvenir à un contrôle total de vos fichiers de commandes, vous avez besoin de développer quelques programmes simples, même si vous n'êtes pas programmeur. La commande DEBUG fournit aux utilisateurs de DOS toutes les fonctions nécessaires à l'écriture de programmes simples destinés à perfectionner les fichiers de commandes.

DEBUG est un «débogueur» ou utilitaire de programmation qui assiste les programmeurs pour la recherche d'erreurs ou «bogues» (*bugs*) dans leurs programmes. Même si, pour la plupart, nous n'emploierons jamais DEBUG pour trouver des erreurs de programmation, nous pouvons l'utiliser pour créer des programmes simples. DEBUG intimide beaucoup de personnes qui le perçoivent comme un utilitaire destiné exclusivement aux programmeurs et nécessitant l'entrée d'instructions en assembleur. Pourtant, si vous suivez les exemples qui vont apparaître dans ce livre, vous découvrirez que DEBUG est d'un emploi facile. Ces exemples expliquent chaque étape de la saisie des programmes. Les quelques minutes passées à écrire ces programmes vont accroître grandement les capacités de vos fichiers de commandes.

DEBUG est une commande DOS externe (sur disque) :

```
C> DIR DEBUG.COM
Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est OF35-18EE
Répertoire de C:\DOS

DEBUG    COM 21606 17/06/88  12:00
      1 Fichier(s) 21166080 octets libres
```

Quand vous lancez DEBUG, vous spécifiez sur la ligne de commande le nom du fichier à créer :

```
C> DEBUG FICHIER.EXT
```

Pour commencer, écrivons un programme IMPREC.COM, qui imprime le contenu actuel de l'écran. Admettons, par exemple, qu'un programme de votre fichier de commandes affiche des données importantes à l'écran. Pour vous assurer que l'opérateur verra ces informations, vous pouvez charger votre fichier de commandes de lancer IMPREC et imprimer ainsi les données affichées à l'écran en fin d'exécution.

Exécutez d'abord DEBUG avec IMPREC.COM comme nom de fichier :

```
C> DEBUG IMPREC.COM
```

DEBUG répondra par :

```
C> DEBUG IMPREC.COM
Fichier introuvable

-
```

Le message *Fichier introuvable* vous confirme que le fichier IMPREC.COM n'existe pas encore sur disque. Le tiret (-) est le message d'attente (indicatif) de DEBUG.

Tapez la commande *A100* et appuyez sur la touche Entrée :

```
C> DEBUG IMPREC.COM
Fichier introuvable
-A 100
5AFF:0100
```

La commande A 100 indique à DEBUG que vous voulez entrer des instructions à l'adresse mémoire correspondant au décalage (*offset*) de valeur hexadécimale 100. Chaque fois que DOS lance un programme, l'exécution commence à cette adresse. C'est donc là que nous placerons nos instructions. Tous les programmes créés avec DEBUG commenceront par la commande A 100.

Examinons la manière dont DEBUG affiche les numéros de ligne de vos programmes. Dans ce cas, le nombre à l'extrême droite est 0100, parce que nous avons choisi de commencer au décalage 100. Le nombre que votre ordinateur affiche à la gauche de 0100 pourrait être différent de celui qui est indiqué dans l'exemple. Ce nombre dépend de l'adresse mémoire où commence DEBUG. Cette adresse peut varier selon votre version de DOS et selon les programmes installés dans l'ordinateur. Tapez la commande *INT 5* et appuyez sur Entrée :

```
C> DEBUG IMPREC.COM
Fichier introuvable

-A 100

5AFF:0100 INT 5
5AFF:0102
```

La commande INT 5 donne à l'ordinateur l'ordre d'imprimer le contenu de l'écran.

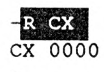
Ce n'est pas plus difficile que cela ! Vous devez tout de même ajouter deux instructions pour que DOS mette fin au programme. En premier lieu, tapez *MOV AH, 4C* et appuyez sur Entrée. En second lieu, tapez *INT 21* et appuyez sur Entrée. Votre programme est terminé.

Il faut maintenant donner à DEBUG l'ordre de sauvegarder sur disque le contenu du programme. Pour initier le processus de sauvegarde, appuyez sur Entrée. DEBUG affiche son message d'attente, comme ci-après :

```
C> DEBUG IMPREC.COM
Fichier introuvable

-A 100
5AFF:0100 INT 5
5AFF:0102 MOV AH, 4C
5AFF:0104 INT 21
5AFF:0106
-
```

Tapez maintenant la commande *R CX* de DEBUG, qui affiche :



```
-R CX
CX 0000
:
```

CX est un registre ou zone de stockage de votre ordinateur. Avant de pouvoir écrire un programme sur disque, DEBUG doit connaître la taille de ce programme. Vous devez donc mettre la taille du programme dans le registre CX. Trouvez la valeur qui correspond au dernier numéro de ligne du programme :

```
C> DEBUG IMPREC.COM
Fichier introuvable

-A 100
5AFF:0100 INT 5
5AFF:0102 MOV AH, 4C
5AFF:0104 INT 21
5AFF:0106————————Dernière ligne du programme
-R CX
CX 0000
:
```

et soustrayez de cette valeur le décalage au départ du programme (par exemple, «0106 moins 0100 égale 6»). Dans ce cas la longueur de votre programme est de six octets. Tapez la valeur 6 après le signe deux-points qui vous invite à entrer la valeur de CX. Quand vous appuyez sur Entrée, DEBUG affiche son indicatif de commande :

```
C> DEBUG IMPREC.COM
Fichier introuvable

-A 100
5AFF:0100 INT 5
5AFF:0102 MOV AH, 4C
5AFF:0104 INT 21
5AFF:0106
-R CX
CX 0000
:6
-
```

Pour sauvegarder le fichier sur disque, vous tapez W, commande Ecrire de DEBUG.

```
C> DEBUG IMPREC.COM
Fichier introuvable

-A 100
5AFF:0100 INT 5
5AFF:0102 MOV AH, 4C
5AFF:0104 INT 21
5AFF:0106
-R CX
CX 0000
:6
-W
Ecriture de 0006 octets
```

Une fois le fichier sauvegardé, utilisez la commande Quitter de DEBUG, Q, pour retourner à l'indicatif DOS :

```
C> DEBUG IMPREC.COM
Fichier introuvable

-A 100
5AFF:0100 INT 5
5AFF:0102 MOV AH, 4C
5AFF:0104 INT 21
5AFF:0106
-R CX
CX 0000
```

```
:6
-W
Ecriture de 0006 octets
-Q

C>
```

Si vous exécutez le programme IMPREC à partir de l'indicatif DOS, DOS imprime le contenu actuel de l'écran. Si vous choisissez la commande DOS DIR avec le nom du fichier IMPREC.COM, DOS affiche :

```
C> DIR IMPREC.COM

Le volume dans l'unité C s'appelle DOS 4.0
Le numéro de série du volume est 0F35-18EE
Répertoire de C:\DOS

IMPREC   COM          6 13/05/90 12:11
      1 fichier(s) 21155840 octets libres
```

Vous voyez que la taille du fichier est de 6 octets.

De façon similaire, le programme RELANCE.COM relance DOS :

```
C> DEBUG RELANCE.COM
Fichier introuvable

-A 100
584B:0100 MOV AX, 40
584B:0103 MOV DS, AX
584B:0105 MOV AX, 1234
584B:0108 MOV MOV [72], AX
584B:010B JMP FFFF:0
584B:0110
-R CX
CX 0000
:10
-W
Ecriture de 0010 octets
-Q

C>
```

RELANCE.COM utilise le programme BOOT qui lance votre ordinateur lorsque vous le mettez sous tension. Ce programme examine la position mémoire 0472H pour voir si elle contient la valeur 1234. Si oui, l'ordinateur effectue un «redémarrage à chaud» du système (du même type que le redémarrage par Ctrl+Alt+Suppr). Le programme RELANCE.COM met la valeur 1234 à l'adresse correcte, puis émet la commande JMP pour effectuer un branchement vers le programme BOOT. Le programme ci-dessous inclut des commentaires expliquant le traitement du programme RELANCE.COM :

```
C> DEBUG RELANCE.COM
Fichier introuvable

-A 100
584B:0100 MOV AX, 40        ; Adresse de
                      segment de 0472H
584B:0103 MOV DS, AX        ; Assigne adresse
                      au segment de données
584B:0105 MOV AX, 1234      ; 1234 lance le
                      programme BOOT
584B:0108 MOV MOV [72], AX ; Place la valeur
                      à son adresse mémoire
584B:010B JMP FFFF:0        ; Branche au
                      programme BOOT
584B:0110
-R CX
CX 0000
:10
-W
Ecriture de 0010 octets
-Q

C>
```

Dans ce cas, si vous exécutez RELANCE à partir de l'indicatif MS-DOS, votre système redémarre. Considérons une application dans laquelle vous auriez besoin d'un fichier de commandes pour relancer DOS. De nombreux utilisateurs emploient des disques virtuels en mémoire pour simuler un

disque dur rapide. Selon votre version de DOS, vous utiliserez le fichier RAMDRIVE.SYS ou le fichier VDISK.SYS pour créer un tel «disque» en mémoire vive. Un disque virtuel ou *RAM drive* est une zone de stockage que vous demandez à DOS de créer dans la mémoire vive ou RAM (*random access memory* : mémoire à accès aléatoire) de votre ordinateur. Une fois ce «disque RAM» créé, les commandes DOS peuvent s'y référer par une simple lettre suivie du signe deux-points, exactement comme dans le cas d'un véritable disque dur ou lecteur de disquettes.

Puisque le lecteur RAM réside dans la mémoire de l'ordinateur, il ne subit aucune des contraintes mécaniques d'un lecteur de disquettes ou d'un disque dur. Il en résulte que le disque RAM est beaucoup plus rapide. Un disque RAM vous donne un lieu de stockage temporaire, à accès très rapide, pour vos fichiers. Toutefois, lorsque vous coupez l'alimentation de l'ordinateur, tout le contenu du disque RAM est perdu.

En admettant que votre version de DOS utilise VDISK.SYS, vous créez un disque RAM en installant le pilote de périphérique VDISK.SYS en mémoire. Servez-vous d' une entrée DEVICE= du fichier CONFIG.SYS. Par exemple, l'entrée :

```
DEVICE=VDISK.SYS
```

crée un disque RAM de 64 Ko (taille par défaut). Pour créer un dique RAM de 256 Ko, utilisez l'entrée :

```
DEVICE=VDISK.SYS 256
```

Remarque : vous devez relancer votre système pour que les modifications de CONFIG.SYS entrent en vigueur.

Le plus souvent, les utilisateurs installent un disque RAM pour des opérations quotidiennes, mais doivent l'enlever quand ils exécutent un programme important qui a besoin de beaucoup de mémoire. Il ne faut pas oublier qu'un disque RAM de 256 Ko occupe 256 Ko de mémoire.

Chaque fois que vous devez enlever le disque RAM de la mémoire, il faut éditer (modifier) le fichier CONFIG.SYS en supprimant l'entrée DEVICE=VDISK.SYS, puis relancer l'ordinateur. Si l'utilisateur veut à nouveau installer le disque RAM, il doit une fois de plus éditer CONFIG.SYS et relancer DOS.

Au lieu de répéter continuellement ce cycle éditer-relancer, vous pouvez créer un fichier de commandes, appelé DISCRAM.BAT, qui accomplit cette tâche à votre place. Par exemple, pour installer le disque RAM, vous exécutez le fichier de commandes comme ceci :

```
C> DISCRAM INSTALL 128
```

Dans ce cas, le fichier de commandes installe un disque RAM de 128 Ko. Pour l'enlever ultérieurement de la mémoire, exécutez le fichier de commandes comme ceci :

```
C> DISCRAM DESINST
```

Dans les deux cas, le fichier de commandes modifie pour vous CONFIG.SYS et relance DOS pour que les modifications entrent en vigueur, comme ceci :

```
ECHO OFF
IF '%1'=='INSTALL' GOTO INSDISC
IF '%1'=='DESINST' GOTO DESDISC
GOTO FIN
:INSDISC
REM Installe le disque RAM en ajoutant la
REM ligne DEVICE=VDISK.SYS dans CONFIG.SYS.
REM Utilise %2 pour déterminer la taille du
REM disque RAM. Une fois CONFIG.SYS modifié,
REM relance le système.
IF NOT EXIST \CONFIG.SYS GOTO AJENTREE
REM
REM Enlève toutes les lignes de CONFIG.SYS
REM contenant VDISK.
REM
TYPE \CONFIG.SYS ¦ FIND /V "VDISK"  >
                                   \CONFIG.NOU
```

```
DEL \CONFIG.SYS
REM
REM Ajoute l'entrée VDISK à CONFIG.SYS
REM
:AJENTREE
ECHO DEVICE=VDISK.SYS %2 >> \CONFIG.NOU
REN \CONFIG.NOU CONFIG.SYS
RELANCE
GOTO FIN

:DESDISC
REM Enlève le disque RAM en supprimant VDISK
REM de CONFIG.SYS et relance DOS.
IF NOT EXIST \CONFIG.SYS GOTO FIN
REM
REM Enlève toutes les lignes de CONFIG.SYS
REM contenant VDISK.
REM
TYPE \CONFIG.SYS ¦ FIND /V "VDISK" >
                                  \CONFIG.NOU
DEL \CONFIF.SYS
REN \CONFIG.NOU CONFIG.SYS
RELANCE
:FIN
```

La commande ECHO qui suit l'étiquette :AJENTREE suppose que le fichier VDISK.SYS se trouve dans le répertoire racine. Si ce n'est pas le cas, vous devez fournir le nom approprié de lecteur et/ou de répertoire en tant que partie intégrante du nom de fichier (par exemple c:\dos\vdisk.sys).

Dans la routine qui commence à l'étiquette :AJENTREE, l'instruction GOTO FIN qui suit l'instruction RELANCE ne doit jamais être exécutée, car le système doit être relancé. Cependant, si le programme RELANCE n'existe pas ou est introuvable, l'instruction GOTO FIN évite que le reste du fichier de commandes soit exécuté accidentellement.

Voici maintenant un programme OUIOUNON.COM assurant l'acquisition des touches frappées par l'utilisateur, jusqu'à ce que celui-ci appuie soit sur O, soit sur N. Quand

OUIOUNON.COM intercepte une touche valide, il renvoie l'information sous forme d'un code de sortie DOS. Il en résulte que votre programme peut utiliser les conditions :

```
OUIOUNON
IF ERRORLEVEL 89 GOTO OUI
IF ERRORLEVEL 78 GOTO NON
```

pour traiter la réponse oui ou non. Vous remarquerez que le fichier de commandes teste d'abord le code de sortie le plus élevé (89).

Rappelez-vous : la condition IF ERRORLEVEL renvoie la valeur «vrai» quand le code de sortie est supérieur ou égal au nombre spécifié. Si vous placiez en première position le test d'un code de sortie de 78, DOS exécuterait toujours les commandes correspondantes car les deux touches renvoient des valeurs supérieures ou égales à 78.

Le programme suivant implémente OUIOUNON.COM :

```
C> DEBUG OUIOUNON.COM
Fichier introuvable
-A100
5B10:0100 MOV AH, 08
5B10:0102 INT 21
5B10:0104 CMP AL, 59 ; Pression sur O ?
5B10:0106 JZ 010E
5B10:0108 CMP AL, 4E ; Pression sur N ?
5B10:010A JZ 010E
5B10:010C JMP 0100   ; Boucle d'attente
                       caractère valide
5B10:010E MOV AH, 4C
5B10:0110 INT 21     ; Fin du programme
5B10:0112
-R CX
CX 0000
:12
-W
Ecriture de 0012 octets
-Q

C>
```

Vous constatez que ce programme est légèrement plus complexe que ceux des deux derniers exemples. Cependant, si vous le reproduisez exactement, vous n'aurez aucun problème.

Pour les programmeurs qui voudraient comprendre le code assembleur utilisé par OUIOUNON.COM, le listage suivant contient une courte explication de chaque ligne :

```
C> DEBUG OUIOUNON.COM
Fichier introuvable

-A100
5B10:0100 MOV AH, 08 ; Service DOS Acquisi-
                           tion caractère
5B10:0102 INT 21     ; Appel du service
5B10:0104 CMP AL, 59 ; Pression sur O ?
5B10:0106 JZ 010E    ; Lettre valide, branche
                           à fin programme
5B10:0108 CMP AL, 4E ; Pression sur N ?
5B10:010A JZ 010E    ; Lettre valide, branche
                           à fin programme
5B10:010C JMP 0100   ; Boucle d'attente
                           caractère valide
5B10:010E MOV AH, 4C ; Service DOS Fin de
                           programme
5B10:0110 INT 21     ; Appel du service, met
                           fin au programme
5B10:0112
-R CX
CX 0000
:12
-W
Ecriture de 0012 octets
-Q

C>
```

Après avoir créé OUIOUNON.COM, vous pouvez le mettre en service immédiatement dans un fichier de commandes appelé DETON.BAT. Ce fichier de commandes emploie les commandes DOS FOR et DEL et vous permet de détruire sélectivement des fichiers. En fait, DETON.BAT crée un

fichier de commandes temporaire qu'il utilise pour détruire les fichiers. Quand le fichier de commandes temporaire, appelé EFFACE.BAT n'est plus nécessaire, DETON.BAT le détruit à son tour :

```
ECHO OFF
REM Crée EFFACE.BAT, fichier de commandes
REM temporaire qui invite l'opérateur à
REM conserver ou à détruire un fichier.
REM Si l'opérateur appuie sur O, le fichier
REM de commandes détruit le fichier.
REM Si l'opérateur appuie sur N, le fichier
REM de commandes conserve le fichier sur le
REM disque.

REM On utilise ECHO et la redirection de DOS
REM pour créer  le fichier de commandes.
ECHO ECHO Voulez-vous détruire %%1? >
                                  EFFACE.BAT
ECHO OUIOUNON >> EFFACE.BAT
ECHO IF ERRORLEVEL 89 DEL %%1 >> EFFACE.BAT

REM Maintenant que le fichier EFFACE.BAT
REM existe sur disque, utilisez-le dans la
REM boucle FOR.

FOR %%I IN (%1) DO CALL EFFACE %%I

REM Le fichier de commandes EFFACE.BAT  n'est
REM plus nécessaire.
DEL EFFACE.BAT
```

Ce fichier fournit beaucoup d'informations. Premièrement, il vous montre que vous pouvez créer un fichier de commandes depuis un autre fichier de commandes et ensuite l'exécuter depuis le fichier primaire. Observons ensuite les trois lignes qui créent le fichier de commandes secondaire :

```
ECHO ECHO Voulez-vous détruire %%1? >
                                  EFFACE.BAT
ECHO OUIOUNON >> EFFACE.BAT
ECHO IF ERRORLEVEL 89 DEL %%1 >> EFFACE.BAT
```

Le fichier de commandes primaire utilise l'opérateur DOS de redirection (>) pour créer le fichier de commandes secondaire EFFACE.BAT. Les deux commandes suivantes emploient l'opérateur DOS de redirection avec ajout (>>) pour intégrer des commandes au fichier EFFACE.BAT. Vous remarquez l'utilisation du double symbole pour cent dans *%%1*.Vous vous souvenez que chaque fois qu'il détecte %1 dans un fichier de commandes, DOS lui substitue la valeur du premier paramètre du fichier.

Dans ce cas, cependant, nous ne voulons pas remplacer un paramètre ; nous voulons que DOS écrive les caractères *%1* dans le fichier de commandes EFFACE.BAT. C'est ce qui se produit lorsque vous écrivez le double signe %. Ainsi, le fichier secondaire comporte des paramètres remplaçables.

Le traitement principal, dans le fichier de commandes DETON.BAT, est la boucle FOR, qui passe chaque nom de fichier au fichier de commandes EFFACE.BAT. Chaque fois que la boucle FOR appelle le fichier de commandes, EFFACE.BAT demande à l'opérateur s'il veut détruire le fichier dont le nom est spécifié. Ensuite, DETON.BAT se sert du programme OUIOUNON.COM, créé précédemment, pour agir selon la réponse de l'opérateur. Si l'opérateur appuie sur O (pour oui), le fichier est détruit. Si l'opérateur appuie sur N (pour non), EFFACE.BAT termine son exécution et la boucle FOR se répète.

Si vous exécutez DETON.BAT comme ceci :

```
C> DETON *.*
```

vous pourrez détruire sélectivement n'importe quel fichier du répertoire courant sur votre lecteur par défaut.

De façon analogue à ce qu'accomplit OUIOUNON.COM, le programme suivant, appelé F1AF10.COM, renvoie les codes clavier de 59 à 68, qui correspondent aux touches F1 à F10. Le programme suivant implémente F1AF10.COM :

```
C> DEBUG F1AF10.COM
Fichier introuvable

-A100
5B10:0100 MOV AH, 08
5B10:0102 INT 21
5B10:0104 CMP AL, 0
5B10:0106 JNZ 0100
5B10:0108 MOV AH, 08
5B10:010A INT 21
5B10:010C CMP AL, 3B
5B10:010E JL 0100
5B10:0110 CMP AL, 44
5B10:0112 JG 0100
5B10:0114 MOV AH, 4C
5B10:0116 INT 21
5B10:0118
-R CX
CX 0000
:18
-W
Ecriture de 0018 octets
-Q

C>
```

Le listage suivant explique chaque étape du traitement :

```
C> DEBUG F1AF10.COM
Fichier introuvable

-A100
5B10:0100 MOV AH, 08 ; Service DOS Acquisi-
                       tion touche
5B10:0102 INT 21     ; Appel du service
5B10:0104 CMP AL, 0  ; AL=0 si touche de
                       fonction
5B10:0106 JNZ 0100   ; Pas zéro, nouvel essai
5B10:0108 MOV AH, 08 ; Service DOS Acquisi-
                       tion code clavier
5B10:010A INT 21     ; Appel du service
5B10:010C CMP AL, 3B ; Compare code clavier
                       à 3B (F1)
```

```
5B10:010E JL 0100    ;  Trop petit, nouvel
                        essai
5B10:0110 CMP AL, 44 ;  Compare code clavier
                        à 44 (F10)
5B10:0112 JG 0100    ;  Trop grand, nouvel
                        essai
5B10:0114 MOV AH, 4C ;  Service DOS Fin de
                        programme
5B10:0116 INT 21     ;  Appelle le service,
                        termine le programme
5B10:0118
-R CX
CX 0000
:18
-W
Ecriture de 0018 octets
-Q

C>
```

Le fichier MENUDOS.BAT affiche le menu suivant :

F1 - Affiche DIR listage d'un répertoire
F2 - Affiche le numéro de version DOS
F3 - Affiche le volume du disque
F4 - Quitte

Si l'utilisateur appuie sur F1, le fichier de commandes affiche la liste des fichiers d'un répertoire. S'il appuie sur F2, le programme affiche le numéro de la version qui est en service. Si l'utilisateur appuie sur F3, il affiche le volume actuel. Le fichier de commandes continue ce traitement jusqu'à ce que l'utilisateur appuie sur F4 pour quitter. Le listage du fichier MENUDOS.BAT est :

```
ECHO OFF
CLS
:BOUCLE
ECHO F1 - Affiche DIR listage d'un répertoire
ECHO F2 - Affiche le numéro de version DOS
ECHO F3 - Affiche le volume du disque
ECHO F4 - Quitte
```

```
REM Acquisition réponse de l'opérateur
:ACQTOUCHE
F1AF10
IF ERRORLEVEL 63 GOTO ACQTOUCHE
IF ERRORLEVEL 62 GOTO FIN
IF ERRORLEVEL 61 IF NOT ERRORLEVEL 62 VOL
IF ERRORLEVEL 60 IF NOT ERRORLEVEL 61 VER
IF ERRORLEVEL 59 IF NOT ERRORLEVEL 60 DIR
GOTO BOUCLE
:FIN
```

Le fichier de commandes commence par afficher le menu puis exécute le programme F1AF10 pour faire l'acquisition de la touche de fonction qui correspond au choix de l'utilisateur. Si celui-ci appuie sur l'une des touches de fonction F5 à F10, le fichier de commandes répète la boucle tant qu'une touche valide n'est pas choisie. Si l'utilisateur appuie sur la touche F4 (code clavier 62), le fichier de commandes met fin au traitement. Remarquez l'emploi des deux commandes IF pour vérifier si un code de sortie a une valeur spécifique. En utilisant ainsi deux commandes IF, le fichier de commandes peut tester si la valeur du code de sortie est égale à 61, 60, ou 59, alors qu'une seule condition testerait une valeur supérieure ou égale à l'un de ces nombres.

Finalement, un programme appelé ACQFLECH.COM attend que l'utilisateur appuie sur la touche Vers le haut, Vers le bas ou Entrée. Le programme renvoie un code de 72 pour la flèche Haut et un code de 80 pour la flèche Bas. Si l'utilisateur appuie sur Entrée, le fichier de commandes renvoie 13. Voici le programme ACQFLECH.COM :

```
C> DEBUG ACQFLECH.COM
Fichier introuvable

-A100
5B10:0100 MOV AH, 08
5B10:0102 INT 21
5B10:0104 CMP AL, 0
5B10:0106 JZ 0100
```

```
5B10:0108 CMP AL, 48
5B10:010A JZ 0116
5B10:010C CMP AL, 50
5B10:010E JZ 0116
5B10:0110 CMP AL, 0D
5B10:0112 JZ 0116
5B10:0114 JMP 0100
5B10:0116 MOV AH, 4C
5B10:0118 INT 21
5B10:011A
-R CX
CX 0000
:1A
-W
Ecriture de 001A octets
-Q

C>
```

Un fichier de commandes FLECHES.BAT affiche un menu similaire à celui que nous avons vu plus haut :

Affiche DIR listage d'un répertoire
Affiche le numéro de version DOS
Affiche le volume du disque
Quitte

Dans ce nouvel exemple, le fichier de commandes emploie le pilote de périphérique ANSI.SYS pour rehausser l'option actuelle choisie sur le menu. Quand l'utilisateur appuie sur la touche (flèche) Haut ou Bas, le fichier de commandes exécute le choix correspondant :

```
ECHO OFF
SET ACTUEL=DIR
:BOUCLE
CLS
IF %ACTUEL%==DIR ECHO Escape[1mAffiche
                 DIR listage d'un répertoire
IF NOT %ACTUEL%==DIR ECHO Escape[0mAffiche
                 DIR listage d'un répertoire
IF %ACTUEL%==VER ECHO Escape[1mAffiche
                 le numéro de version DOS
```

```
IF NOT %ACTUEL%==VER ECHO Escape[0mAffiche
                 le numéro de version DOS
IF %ACTUEL%==VOL ECHO Escape[1mAffiche
                 le volume du disque
IF NOT %ACTUEL%==VOL ECHO Escape[0mAffiche
                 le volume du disque
IF %ACTUEL%==QUIT ECHO Escape[1mQuitte
IF NOT %ACTUEL%==QUIT ECHO Escape[0mQuitte
ACQFLECH
IF ERRORLEVEL 80 GOTO BAS_FL
IF ERRORLEVEL 72 GOTO HAUT_FL
IF %ACTUEL%==DIR DIR
IF %ACTUEL%==VER VER
IF %ACTUEL%==VOL VOL
IF %ACTUEL%==QUIT GOTO FIN
PAUSE
GOTO BOUCLE

:HAUT_FL
IF NOT %ACTUEL%==DIR GOTO HAUT__VER
SET ACTUEL=QUIT
GOTO BOUCLE
:HAUT__VER
IF NOT %ACTUEL%==VER GOTO HAUT__VOL
SET ACTUEL=DIR
GOTO BOUCLE
:HAUT__VOL
IF NOT %ACTUEL%==VOL GOTO HAUT__QUIT
SET ACTUEL=VER
GOTO BOUCLE
:HAUT__QUIT
SET ACTUEL=VOL
GOTO BOUCLE

:BAS_FL
IF NOT %ACTUEL%==DIR GOTO BAS__VER
SET ACTUEL=VER
GOTO BOUCLE
:BAS__VER
IF NOT %ACTUEL%==VER GOTO BAS__VOL
SET ACTUEL=VOL
GOTO BOUCLE
```

```
:BAS__VOL
IF NOT %ACTUEL%==VOL GOTO BAS__QUIT
SET ACTUEL=QUIT
GOTO BOUCLE
:BAS__QUIT
SET ACTUEL=DIR
GOTO BOUCLE

:FIN
ECHO Escape[0m
SET ACTUEL=
```

Les capacités du fichier de commandes à la manipulation de menus sont normalement réservées aux programmes écrits dans des langages tels que Pascal ou C. Le fichier de commandes de l'exemple ci-dessus suit la trace de l'option actuelle, en utilisant le paramètre nommé ACTUEL. Quand le fichier de commandes commence son exécution, il affecte à l'aide de la commande SET la valeur DIR au paramètre nommé. Si vous regardez le menu que FLECHES.BAT affiche, DIR est la première des options. Ensuite, le fichier de commandes se sert de la commande de traitement par lots IF pour déterminer l'option actuelle. Le fichier de commandes affiche l'option actuelle avec un attribut gras, grâce à la séquence d'échappement ANSI.SYS *Escape[1m*. Pour afficher toutes les autres options avec l'attribut normal, le fichier de commandes emploie la séquence d'échappement *Escape[0m*.

Le fichier de commandes utilise ensuite le programme ACQFLECH.COM, récemment créé, pour faire l'acquisition de la touche Vers le haut, de la touche Vers le bas ou de la touche Entrée. Si l'utilisateur appuie sur Entrée, le fichier de commandes se sert du paramètre nommé ACTUEL pour déterminer la commande à exécuter. Si l'utilisateur appuie soit sur la touche Vers le haut, soit sur la touche Vers le bas, le fichier de commandes détermine la nouvelle option actuelle et réaffiche le menu.

Quand l'utilisateur choisit finalement la commande Quitte, le fichier de commandes réinitialise l'attribut vidéo à la valeur normale et supprime le paramètre nommé ACTUEL de l'environnement DOS.

Il est vrai que les options de notre menu sont très rudimentaires ; cependant, FLECHES.BAT illustre l'ampleur de la programmation qui peut être effectuée grâce aux fichiers de commandes.

UTILISER LE JEU ÉTENDU DE CARACTÈRES ASCII

L'ordinateur représente tous les caractères, chiffres et symboles affichés par DOS par des valeurs distinctes comprises entre 0 et 255. Les 128 premières valeurs (de 0 à 127) correspondent aux symboles les plus couramment utilisés : chiffres, lettres et signes de ponctuation. Ces 128 valeurs sont connues sous le nom de jeu standard des caractères ASCII. Les valeurs 128 à 255 sont connues sous l'appellation de jeu étendu des caractères ASCII. Ces caractères comprennent des symboles semi-graphiques pour l'ordinateur personnel IBM PC et pour les ordinateurs compatibles. Le tableau de la page suivante contient le jeu étendu de caractères ; pour chaque symbole S, il donne la valeur correspondante du code ASCII.

Au fur et à mesure des perfectionnements apportés à vos fichiers de commandes, vous aurez les moyens de rehausser vos affichages écran en encadrant les menus ou autres informations.

Considérons un fichier de commandes CADMENU.BAT, qui affiche dans un cadre le menu que nous avons déjà pris comme exemple. L'illustration de la page 164 représente d'une part le menu encadré et d'autre part les caractères utilisés pour dessiner le cadre.

S	ASCII	S	ASCII	S	ASCII	S	ASCII
Ç	128	á	160	└	192	α	224
ü	129	í	161	┴	193	β	225
é	130	ó	162	┬	194	Γ	226
â	131	ú	163	├	195	π	227
ä	132	ñ	164	─	196	Σ	228
à	133	Ñ	165	┼	197	σ	229
å	134	ª	166	╞	198	µ	230
ç	135	º	167	╟	199	τ	231
ê	136	¿	168	╚	200	Φ	232
ë	137	⌐	169	╔	201	Θ	233
è	138	¬	170	╩	202	Ω	234
ï	139	½	171	╦	203	δ	235
î	140	¼	172	╠	204	∞	236
ì	141	¡	173	═	205	∅	237
Ä	142	«	174	╬	206	ε	238
Å	143	»	175	╧	207	∩	239
É	144	░	176	╨	208	≡	240
æ	145	▒	177	╤	209	±	241
Æ	146	▓	178	╥	210	≥	242
ô	147	│	179	╙	211	≤	243
ö	148	┤	180	╘	212	⌠	244
ò	149	╡	181	╒	213	⌡	245
û	150	╢	182	╓	214	÷	246
ù	151	╖	183	╫	215	≈	247
ÿ	152	╕	184	╪	216	°	248
Ö	153	╣	185	┘	217	•	249
Ü	154	║	186	┌	218	·	250
¢	155	╗	187	█	219	√	251
£	156	╝	188	▄	220	n	252
¥	157	╜	189	▌	221	^{2}	253
Pt	158	╛	190	▐	222	■	254
ƒ	159	┐	191	▀	223	<blanc>	255

```
F1 - Affiche DIR listage d'un répertoire
F2 - Affiche le numéro de version DOS
F3 - Affiche le volume du disque
F4 - Quitte
```

Pour tracer le cadre, le fichier de commandes utilise les caractères du jeu étendu ayant pour valeurs du code ASCII 179, 191, 192, 196, 217 et 218, comme ceci :

```
┌ 218        196        191 ┐
│ 179                   179 │
└ 192        196        217 ┘
```

C'est avec Edlin que vous allez créer le plus facilement ce fichier de commandes. Commencez par taper les commandes ECHO OFF et CLS ainsi que l'étiquette :BOUCLE.

```
C> EDLIN CADMENU.BAT
Nouveau fichier
*I
        1:* ECHO OFF
        2:* CLS
        3:* :BOUCLE
        4:*
```

Entrez ensuite ECHO, mais sans appuyer sur Entrée. Maintenez la touche Alt enfoncée et tapez *218* (en utilisant le pavé numérique à l'extrême droite de votre clavier). Quand vous relâchez la touche Alt, l'écran affiche l'angle supérieur gauche du cadre.

```
C> EDLIN CADMENU.BAT
Nouveau fichier
*I
```

```
        1:* ECHO OFF
        2:* CLS
        3:* :BOUCLE
        4:* ECHO ┌
```

En maintenant à nouveau la touche Alt enfoncée, tapez *196*.

Quand vous relâchez Alt, DOS affiche un segment de droite horizontale qui prolonge la partie supérieure (horizontale) du cadre :

```
C> EDLIN CADMENU.BAT
Nouveau fichier
*I
        1:* ECHO OFF
        2:* CLS
        3:* :BOUCLE
        4:* ECHO ┌─
```

Répétez ce processus et frappez un total de 44 segments horizontaux :

```
C> EDLIN CADMENU.BAT
Nouveau fichier
*I
        1:* ECHO OFF
        2:* CLS
        3:* :BOUCLE
        4:* ECHO ┌────────────────────────────────────────────
```

Pour terminer le haut du cadre, c'est-à-dire pour lui ajouter l'angle supérieur droit, maintenez la touche Alt enfoncée et tapez le nombre *191* :

```
C> EDLIN CADMENU.BAT
Nouveau fichier
*I
        1:* ECHO OFF
        2:* CLS
        3:* :BOUCLE
        4:* ECHO ┌────────────────────────────────────────────┐
```

A l'aide du caractère de code ASCII 179 pour dessiner un segment de droite verticale, terminez les sept lignes suivantes comme ceci :

```
 C> EDLIN CADMENU.BAT
Nouveau fichier
*I
        1:* ECHO OFF
        2:* CLS
        3:* :BOUCLE
        4:* ECHO ┌──────────────────────────────────────┐
        5:* ECHO │ F1 - Affiche DIR listage d'un répertoire │
        6:* ECHO │                                      │
        7:* ECHO │ F2 - Affiche le numéro de version DOS │
        8:* ECHO │                                      │
        9:* ECHO │ F3 - Affiche le volume du disque     │
       10:* ECHO │                                      │
       11:* ECHO │ F4 - Quitte                          │
```

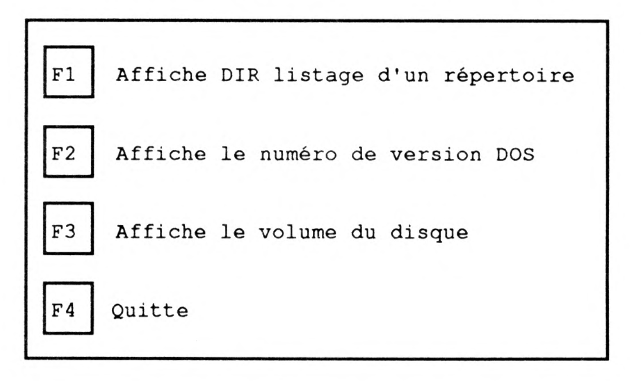
La partie inférieure du cadre ressemble beaucoup à la partie supérieure, mais il faut utiliser les caractères de codes 192 et 217 pour les angles inférieur gauche et inférieur droit du cadre.

Entrez les commandes du menu MENUDOS.BAT, que vous avez créé précédemment. CADMENU.BAT devient :

```
C> EDLIN CADMENU.BAT
Nouveau fichier
*I
        1:* ECHO OFF
        2:* CLS
        3:* :BOUCLE
        4:* ECHO ┌──────────────────────────────────────┐
        5:* ECHO │ F1 - Affiche DIR listage d'un répertoire │
        6:* ECHO │                                      │
        7:* ECHO │ F2 - Affiche le numéro de version DOS │
        8:* ECHO │                                      │
        9:* ECHO │ F3 - Affiche le volume du disque     │
       10:* ECHO │                                      │
       11:* ECHO │ F4 - Quitte                          │
       12:* ECHO └──────────────────────────────────────┘
       13:* REM Acquisition réponse opérateur
       14:* :ACQTOUCHE
       15:* F1AF10
       16:* IF ERRORLEVEL 63 GOTO ACQTOUCHE
       17:* IF ERRORLEVEL 62 GOTO FIN
```

```
18:* IF ERRORLEVEL 61 IF NOT
                         ERRORLEVEL 62 VOL
19:* IF ERRORLEVEL 60 IF NOT
                         ERRORLEVEL 61 VER
20:* IF ERRORLEVEL 59 IF NOT
                         ERRORLEVEL 60 DIR
21:* GOTO BOUCLE
22:* :FIN
23:* ^C
```

Le jeu étendu de caractères ASCII ouvre considérablement l'éventail des présentations possibles à l'écran. N'hésitez pas à soigner vos écrans. Par exemple, en ajoutant quelques commandes ECHO au fichier ci-dessus, le menu devient :



A présent, vous avez assemblé une puissante collection d'outils de traitement par lots. Faites preuve de vos qualifications nouvelles en développant des écrans de présentation professionnelle.

CHAPITRE 8

Votre bibliothèque de fichiers de commandes

DÉTERMINER LES COULEURS DE L'ÉCRAN

Le fichier de commandes COULEURS.BAT vous permet d'utiliser les touches de direction pour parcourir le cycle complet des couleurs d'arrière-plan reconnues par le pilote de périphérique ANSI.SYS. Chaque fois que vous appuyez sur l'une des touches (flèches) Vers le haut ou Vers le Bas, le fichier de commandes change la couleur d'écran. Appuyez sur Entrée et la couleur que vous venez de sélectionner restera active pour la durée de votre session de travail.

```
ECHO OFF
REM Permettre à l'utilisateur de parcourir
REM les couleurs d'arrière-plan disponibles
REM et d'appuyer sur Entrée pour sélectionner
REM une couleur
SET ACTUEL=NOIR
:BOUCLE
IF %ACTUEL%==NOIR ECHO Escape[40m
IF %ACTUEL%==ROUGE ECHO Escape[41m
IF %ACTUEL%==VERT ECHO Escape[42m
IF %ACTUEL%==JAUNE ECHO Escape[43m
IF %ACTUEL%==BLEU ECHO Escape[44m
IF %ACTUEL%==MAGENTA ECHO Escape[45m
```

```
IF %ACTUEL%==CYAN ECHO Escape[46m
IF %ACTUEL%==BLANC ECHO Escape[47m
CLS
ECHO Appuyez sur Entrée pour sélectionner la
ECHO couleur actuelle
ECHO Utilisez les Flèches Haut et Bas pour
ECHO changer de couleur.
ACQFLECH
IF ERRORLEVEL 80 GOTO BAS_FL
IF ERRORLEVEL 72 GOTO HAUT_FL
IF ERRORLEVEL 13 GOTO FIN
:BAS_FL
IF NOT %ACTUEL%==NOIR GOTO BAS_ROUGE
SET ACTUEL=ROUGE
GOTO BOUCLE
:BAS_ROUGE
IF NOT %ACTUEL%==ROUGE GOTO BAS_VERT
SET ACTUEL=VERT
GOTO BOUCLE
:BAS_VERT
IF NOT %ACTUEL%==VERT GOTO BAS_JAUNE
SET ACTUEL=JAUNE
GOTO BOUCLE
:BAS_JAUNE
IF NOT %ACTUEL%==JAUNE GOTO BAS_BLEU
SET ACTUEL=BLEU
GOTO BOUCLE
:BAS_BLEU
IF NOT %ACTUEL%==BLEU GOTO BAS_MAGENTA
SET ACTUEL=MAGENTA
GOTO BOUCLE
:BAS_MAGENTA
IF NOT %ACTUEL%==MAGENTA GOTO BAS_CYAN
SET ACTUEL=CYAN
GOTO BOUCLE
:BAS_CYAN
IF NOT %ACTUEL%==CYAN GOTO BAS_BLANC
SET ACTUEL=BLANC
GOTO BOUCLE
:BAS_BLANC
SET ACTUEL=NOIR
GOTO BOUCLE
```

```
:HAUT_FL
IF NOT %ACTUEL%==NOIR GOTO HAUT_ROUGE
SET ACTUEL=BLANC
GOTO BOUCLE
:HAUT_ROUGE
IF NOT %ACTUEL%==ROUGE GOTO HAUT_VERT
SET ACTUEL=NOIR
GOTO BOUCLE
:HAUT_VERT
IF NOT %ACTUEL%==VERT GOTO HAUT_JAUNE
SET ACTUEL=ROUGE
GOTO BOUCLE
:HAUT_JAUNE
IF NOT %ACTUEL%==JAUNE GOTO HAUT_BLEU
SET ACTUEL=VERT
GOTO BOUCLE
:HAUT_BLEU
IF NOT %ACTUEL%==BLEU GOTO HAUT_MAGENTA
SET ACTUEL=JAUNE
GOTO BOUCLE
:HAUT_MAGENTA
IF NOT %ACTUEL%==MAGENTA GOTO HAUT_CYAN
SET ACTUEL=BLEU
GOTO BOUCLE
:HAUT_CYAN
IF NOT %ACTUEL%==CYAN GOTO HAUT_BLANC
SET ACTUEL=MAGENTA
GOTO BOUCLE
:HAUT_BLANC
SET ACTUEL=CYAN
GOTO BOUCLE

:FIN
```

Bien que long, le fichier de commandes COULEURS.BAT est simple. Il utilise le paramètre nommé ACTUEL pour suivre la trace de la couleur actuelle d'arrière-plan. Selon la valeur d'ACTUEL, le fichier de commandes envoie à l'écran la séquence d'échappement ANSI.SYS correspondante. Le fichier de commandes emploie ACQFLECH.COM, programme que nous avons créé précédemment avec DEBUG.

Si vous appuyez sur l'une des touches (flèches) Haut ou Bas, le fichier de commandes parcourt les différentes couleurs jusqu'à celle qui vous convient. Dès que vous appuyez sur Entrée, le traitement cesse. Comme nous l'avons déjà indiqué, pour que ce fichier de commandes fonctionne, il est nécessaire d'installer au préalable le pilote de périphérique ANSI.SYS dans le fichier CONFIG.SYS.

ÉVITER DE FORMATER LE DISQUE DUR

Bien des disques durs ont été les victimes de commandes FORMAT erronées émises par des utilisateurs novices. Pour prévenir ce genre de désagrément, on supprime souvent le fichier FORMAT.COM d'un disque dur partagé par des utilisateurs multiples. Cette pratique élimine la possibilité de formater accidentellement le disque dur, mais elle prive également d'une commande DOS indispensable. Des gens qui font un travail sérieux sur un système à disque dur auront sans aucun doute besoin de copier des données sur des disquettes. Il faudra donc bien qu'un jour ou l'autre ils formatent des disquettes.

Une solution bien préférable consiste à renommer la commande FORMAT de DOS que l'on appelle, par exemple, FMAT.COM. On crée ensuite le fichier de commandes FORMAT.BAT suivant :

```
ECHO OFF
IF '%1'=='' GOTO ABS_PARAM
ECHO Pour éviter le formatage accidentel du
ECHO disque dur, ce fichier de commandes
ECHO ne permet de formater que les disquettes
ECHO dans le lecteur A ou B. Spécifiez le
ECHO lecteur dans votre ligne de commande.
ECHO Exemple : FORMAT A:
ECHO Voulez-vous continuer (O/N) ?
OUIOUNON
```

```
IF ERRORLEVEL 89 GOTO CONTINUE
IF ERRORLEVEL 78 GOTO FIN
:CONTINUE
IF '%1'=='' GOTO ABS_PARAM
IF '%1'=='A:' GOTO LECT_VALIDE
IF '%1'=='B:' GOTO LECT_VALIDE
GOTO LECT_INVALIDE
:ABS_PARAM
ECHO Vous devez spécifier le lecteur à
ECHO formater dans votre ligne de commande.
GOTO FIN
:LECT_VALIDE
FMAT %1 %2 %3 %4 %5 %6 %7 %8 %9
GOTO FIN
:LECT_INVALIDE
ECHO Ce fichier de commandes ne formate que
ECHO les disquettes dans le lecteur A ou B.
ECHO Il considère le lecteur %1 comme non
ECHO valide.
:FIN
```

Quand vous exécutez FORMAT.BAT, DOS affiche :

```
Pour éviter le formatage accidentel du
disque dur, ce fichier de commandes
ne permet de formater que les disquettes
dans le lecteur A ou B. Spécifiez le
lecteur dans votre ligne de commande.
Exemple : FORMAT A:
Voulez-vous continuer (O/N) ?
```

Si vous tapez O pour continuer le fichier de commandes, il vérifie si vous tentez de formater un disque dans un lecteur autre que A ou B. Si oui, le fichier de commandes affiche un message d'erreur et cesse le traitement. Si vous formatez un disque dans le lecteur A ou B, FMAT.COM est exécuté en utilisant tous les paramètres remplaçables.

Rappelez-vous : dans certains cas, la ligne de commande FORMAT exige des paramètres additionnels, comme /S ou /4. Grâce aux paramètres remplaçables %2 à %9, le fichier de commandes reconnaît ces paramètres supplémentaires.

ACQUÉRIR UNE CHAÎNE DE CARACTÈRES

En suivant les exemples de ce livre, vous avez créé, grâce à DEBUG, des programmes permettant à l'utilisateur d'appuyer sur O ou N pour répondre par oui ou par non, d'entrer une touche de fonction spécifique et d'appuyer sur les touches de direction Vers le haut et Vers le bas. Dans certains cas, cependant, vous pouvez avoir besoin d'intercepter plusieurs caractères frappés par l'utilisateur, comme par exemple un nom de fichier ou un mot de passe contrôlant l'accès à un fichier de commandes.

Le fichier de commandes suivant, appelé ACMODPAS.BAT, invite l'utilisateur à entrer un mot de passe. Ce fichier de commandes compare ensuite le mot de passe entré au mot mot de passe prédéfini AUTORISATION. Si les deux mots de passe sont identiques, le fichier de commandes continue le traitement. Sinon, il affiche un message d'erreur et l'exécution s'arrête :

```
ECHO OFF
IF NOT EXIST MODPAS.DEF GOTO ABSENT
ECHO Entrez votre mot de passe, appuyez sur
ECHO F6, puis sur Entrée.
COPY CON MODPAS.DON > NUL
COPY MODPAS.DEF+MODPAS.DON MODPAS.BAT > NUL
CALL MODPAS
IF NOT '%MODPAS%'=='AUTORISATION' GOTO
                                        MAUV_MODPAS
ECHO Mot de passe valide.
ECHO Prêt à traiter la paye.
GOTO FIN
:ABSENT
ECHO Ce fichier de commandes exige le fichier
ECHO MODPAS.DEF, qui contient SET MODPAS=^Z,
ECHO comme expliqué dans le livre de
ECHO référence des fichiers de commandes.
GOTO FIN
:MAUV_MODPAS
```

```
ECHO Mot de passe non valide
:FIN
SET MODPAS=
DEL MODPAS.BAT
DEL MODPAS.DON
```

Le fichier de commandes fait appel à MODPAS.DEF, fichier que vous pouvez créer après le message d'attente de DOS. Pour commencer, tapez la commande ci-dessous :

```
C> COPY CON MODPAS.DEF
```

Tapez ensuite *SET MODPAS=*, mais au lieu d'appuyer sur Entrée, appuyez sur F6 immédiatement après le signe égal, puis sur Entrée. DOS crée alors le fichier MODPAS.DEF, comme ceci :

```
C> COPY CON MODPAS.DEF
SET MODPAS=^Z
        1 Fichier(s) copié(s)

C>
```

Quand vous exécutez ACMODPAS.BAT, vous êtes invité par le fichier de commandes à entrer un mot de passe :

```
Entrez votre mot de passe, appuyez sur
F6, puis sur Entrée.
```

A cette invitation, tapez *AUTORISATION*, appuyez sur F6, puis sur Entrée. Le fichier de commandes copie alors les informations que vous tapez dans le fichier MODPAS.DON. La commande :

```
COPY CON MODPAS.DON > NUL
```

donne à DOS l'ordre de copier les informations entrées au clavier dans le fichier MODPAS.DON, jusqu'au moment où vous appuyez sur F6 pour indiquer la fin de fichier. La commande redirige le message :

```
        1 Fichier(s) copié(s)
```

vers le périphérique NUL. Ainsi, ce message n'apparaît pas à l'écran.

Ensuite, la commande :

```
COPY MODPAS.DEF+MODPAS.DON MODPAS.BAT > NUL
```

donne à DOS l'ordre d'ajouter le contenu de MODPAS.DON au contenu du fichier MODPAS.DEF, pour créer le fichier MODPAS.BAT. En supposant que vous ayez tapé le mot de passe *AUTORISATION*, le fichier MODPAS.BAT contient maintenant :

```
SET MODPAS=AUTORISATION
```

ACMODPAS exécute MODPAS.BAT, fichier de commandes qui crée le paramètre nommé MODPAS dans l'environnement DOS. Dès lors que cette entrée existe, le fichier de commandes peut comparer le mot de passe avec le mot AUTORISATION en utilisant la commande de traitement par lots IF. Si les mots de passe diffèrent, le fichier de commandes affiche un message d'erreur et arrête le traitement. Si les mots de passe sont identiques, le fichier de commandes peut exécuter ses commandes protégées. Vous noterez qu'avant de terminer son exécution, le fichier de commandes supprime l'entrée MODPAS de l'environnement DOS et détruit MODPAS.BAT et MODPAS.DON.

Certes, il y a peu de chances pour que vous ayez besoin d'un mot de passe pour la plupart de vos applications des fichiers de commandes. Le but de cet exemple était de vous montrer comment faire une acquisition de données utilisateur à partir d'un fichier de commandes.

SIMPLIFIER LES OPÉRATIONS DE SAUVEGARDE DU DISQUE DUR

Un fichier de commandes appelé SAUVGARD.BAT vous aide à effectuer les opérations de sauvegarde du disque dur. Quand vous exécutez ce fichier de commandes, DOS affiche le menu suivant :

```
F1 - Sauvegarde mensuelle
F2 - Sauvegarde journalière
F3 - Sauvegarde spécifique
F4 - Quitte
```

Si vous appuyez sur la touche de fonction F1, le fichier de commandes effectue une sauvegarde mensuelle ou sauvegarde système, c'est-à-dire une sauvegarde de tous les fichiers du disque dur. Selon le nombre de fichiers que contient votre disque, ce type de sauvegarde peut demander beaucoup de temps. Si vous appuyez sur la touche de fonction F2, le fichier de commandes fera une sauvegarde journalière ou sauvegarde différentielle, n'enregistrant que les fichiers créés ou modifiés depuis la dernière sauvegarde (ce type d'opération est souvent beaucoup plus rapide que la sauvegarde du système). Vous devez cependant garder tous vos disques de sauvegarde différentielle entre une sauvegarde système et la suivante. Ainsi, vous êtes sûr d'avoir une copie de tous les fichiers du disque dur.

Si vous appuyez sur la touche de fonction F3, le fichier de commandes vous invite à entrer les spécifications des fichiers à sauvegarder, ainsi que le lecteur de destination. Pour sauvegarder tous les fichiers du sous-répertoire DOS vers la disquette du lecteur A, par exemple, il est nécessaire de taper :

```
C:\DOS  A:
```

Vous tapez ensuite la touche de fonction F6 et appuyez sur Entrée. Le fichier de commandes sauvegarde seulement les fichiers spécifiés.

Si vous appuyez sur la touche F4, Le traitement cesse.

Pour une compréhension plus aisée, considérez ce fichier de commandes comme l'ensemble de quatre fichiers de commandes distincts : SAUVGARD.BAT, MENSUEL.BAT, JOURNAL.BAT, et SPECIF.BAT. Le fichier de commandes SAUVGARD.BAT contient :

```
ECHO OFF
REM Permet d'effectuer une sauvegarde
REM mensuelle, journalière ou spécifique.
CLS
REM La boucle se répète jusqu'au moment où
REM l'utilisateur appuie sur F4 pour quitter.
:BOUCLE
ECHO F1 - Sauvegarde mensuelle
ECHO F2 - Sauvegarde journalière
ECHO F3 - Sauvegarde spécifique
ECHO F4 - Quitte
REM Acquisition de la réponse utilisateur.
:ACQTOUCH
F1AF10
REM Si l'utilisateur appuie sur une touche
REM autre que F1 à F4, acquisition d'une
REM autre touche.
IF ERRORLEVEL 63 GOTO ACQTOUCH
IF ERRORLEVEL 62 GOTO FIN
IF ERRORLEVEL 61 IF NOT ERRORLEVEL 62 CALL
                                     SPECIF
IF ERRORLEVEL 60 IF NOT ERRORLEVEL 61 CALL
                                     JOURNAL
IF ERRORLEVEL 59 IF NOT ERRORLEVEL 60 CALL
                                     MENSUEL
GOTO BOUCLE

:FIN
```

Vous constatez que ce fichier de commandes affiche le menu principal, puis utilise le programme F1AF10.COM pour intercepter la pression d'une touche de fonction par l'utilisateur. Selon la touche de fonction frappée, le fichier de commandes appelle MENSUEL.BAT pour une sauvegarde complète du disque, JOURNAL.BAT pour une sauvegarde différentielle ou SPECIF.BAT pour la sauvegarde de fichiers spécifiques. Le fichier de commandes répète la boucle jusqu'à ce que l'utilisateur appuie sur la touche F4 pour quitter.

Le fichier de commandes MENSUEL.BAT effectue une sauvegarde complète du disque. Quand vous sélectionnez cette option, DOS affiche :

```
La sauvegarde mensuelle copie sur
disquettes tous les fichiers de votre
disque dur.

Selon le nombre de fichiers sur votre
disque, la sauvegarde mensuelle peut
éventuellement être très longue.
Voulez-vous continuer (O/N) ?
```

Ce message explique simplement les procédures de sauvegarde mensuelle et permet à l'utilisateur de continuer ou d'annuler l'opération. Si l'utilisateur répond en appuyant sur O pour continuer, la commande BACKUP de DOS copie la totalité du disque sur des disquettes insérées successivement dans le lecteur A :

```
REM Effectue une sauvegarde complète du
REM disque du système.
ECHO La sauvegarde mensuelle copie sur
ECHO disquettes tous les fichiers de votre
ECHO disque dur.

ECHO Selon le nombre de fichiers sur votre
ECHO disque, la sauvegarde mensuelle peut
ECHO éventuellement être très longue.
ECHO Voulez-vous continuer (O/N) ?
OUIOUNON
IF ERRORLEVEL 89 GOTO SAUVE
IF ERRORLEVEL 78 GOTO RETOUR
:SAUVE
ECHO Prenez soin d'étiqueter chaque disquette
ECHO avec date du jour, vos initiales et
ECHO le titre SAUVEGARDE MENSUELLE.
BACKUP C:\*.* A: /S
:RETOUR
CLS
```

Le fichier de commandes JOURNAL.BAT est très similaire au fichier de commandes MENSUEL.BAT :

```
REM Effectue une sauvegarde de chaque
REM fichier du disque créé ou modifié depuis
REM la dernière sauvegarde.

ECHO La sauvegarde journalière copie sur
ECHO disquettes tous les fichiers de votre
ECHO disque dur qui ont été créés ou modifiés
ECHO depuis la dernière sauvegarde.
ECHO
ECHO Voulez-vous continuer (O/N) ?
OUIOUNON
IF ERRORLEVEL 89 GOTO SAUVE
IF ERRORLEVEL 78 GOTO RETOUR
:SAUVE
ECHO Prenez soin d'étiqueter chaque disquette
ECHO avec la date du jour, vos initiales et
ECHO le titre SAUVEGARDE JOURNALIERE.
ECHO Utilisez la disquette de la précédente
ECHO sauvegarde journalière, jusqu'à ce
ECHO qu'elle soit pleine.
BACKUP C:\*.* A: /S /A /M
:RETOUR
CLS
```

Vous voyez que si l'utilisateur répond en appuyant sur O pour continuer le fichier de commandes, la commande BACKUP exécute une sauvegarde différentielle des fichiers du disque dur.

Des quatre fichiers de commandes destinés à la sauvegarde, le plus intéressant est le fichier SPECIF.BAT. Ce fichier de commandes permet à l'utilisateur de taper les noms des fichiers à sauvegarder. SPECIF.BAT a recours à un fichier appelé SAUVEG.FMT, que vous pouvez créer après l'indicatif DOS. Le fichier SPECIF.BAT contient :

```
REM Sauvegarde sur disquettes un ou
REM plusieurs fichiers spécifiés par
REM l'utilisateur. Invite l'utilisateur à
REM entrer le nom du fichier à sauvegarder.
```

```
REM Copie les noms du ou des fichiers vers le
REM fichier FICASAUV.DON. Crée, à l'aide du
REM fichier SAUVEG.FMT, une ligne de commande
REM BACKUP dans le fichier de commandes
REM SAUVEG.BAT.
IF NOT EXIST SAUVEG.FMT GOTO ABSENT
ECHO L'option fichiers spécifiques vous
ECHO permet de sauvegarder  un ou plusieurs
ECHO fichiers.
ECHO Voulez-vous continuer (O/N) ?
OUIOUNON
IF ERRORLEVEL 89 GOTO SAUVE
IF ERRORLEVEL 78 GOTO RETOUR
:ABSENT
ECHO La sauvegarde de fichiers spécifiques
ECHO exige le fichier SAUVEG.FMT, qui
ECHO contient BACKUP ^Z (explications dans le
ECHO Quick Reference Fichiers Batch MS-DOS)
PAUSE
GOTO RETOUR
:SAUVE
ECHO Tapez la spécification des fichiers à
ECHO sauvegarder et l'identification du
ECHO lecteur de disquettes de destination,
ECHO appuyez sur F6, puis sur Entrée.
ECHO Exemple : *.DON A: ^Z [Entrée]
COPY CON FICASAUV.DON > NUL
COPY SAUVEG.FMT+FICASAUV.DON SAUVEG.BAT
CALL SAUVEG
DEL FICASAUV.DON
DEL SAUVEG.BAT
:RETOUR
CLS
```

A l'aide de la commande COPY CON SAUVEG.FMT, commencez la copie du fichier :

```
C> COPY CON SAUVEG.FMT
```

Tapez *BACKUP* suivi d'un espace et appuyez sur F6 pour marquer la fin du fichier :

```
C> COPY CON  SAUVEG.FMT
BACKUP ^Z
```

Quand vous appuyez sur Entrée, DOS crée le fichier et votre écran affiche maintenant :

```
C> COPY CON  SAUVEG.FMT
BACKUP ^Z
        1 Fichier(s) copié(s)

C>
```

Si le fichier de commandes découvrait un jour que le fichier SAUVEG.FMT n'existe pas, il afficherait un message d'erreur et retournerait au menu principal.

En supposant que SAUVEG.FMT existe, si vous appuyez sur O pour continuer le fichier de commandes, le fichier de commandes vous invite à taper le nom de fichier à sauvegarder et le lecteur contenant la disquette de destination :

```
Tapez la spécification des fichiers à
sauvegarder et l'identification du
lecteur de disquettes de destination,
appuyez sur F6, puis sur Entrée.
Exemple : *.DON A: ^Z [Entrée]
```

Le fichier de commandes place les informations que vous entrez dans le fichier FICASAUV.DON. En redirigeant sa sortie vers le périphérique NUL, la commande :

```
C> COPY CON  FICASAUV.DON > NUL
```

supprime l'affichage du message :

```
        1 Fichier(s) copié(s)
```

La commande :

```
COPY SAUVEG.FMT+FICASAUV.DON SAUVEG.BAT
```

ajoute le contenu du fichier FICASAUV.DON (qui contient le nom du fichier à sauvegarder) au fichier SAUVEG.FMT (qui contient la commande BACKUP). Le fichier résultant est un fichier de commandes appelé SAUVEG.BAT, qui est alors exécuté par SPECIF.BAT. En supposant que l'utilisateur tape *C:\DOS*, puis *A:*, SAUVEG.BAT contient :

```
BACKUP C:\DOS A:
```

Vous constatez que la commande sauvegarde seulement le sous-répertoire désiré. Lorsque l'exécution de la commande BACKUP est terminée, le fichier de commandes détruit les fichiers FICASAUV.DON et SAUVEG.BAT, éliminant du disque tout fichier devenu maintenant inutile.

La collection des fichiers de commandes effectuant les diverses opérations de sauvegarde vous montre combien il est facile de développer des utilitaires puissants fondés sur les fichiers de commandes de DOS. Si vous vouliez perfectionner ces fichiers de commandes, vous pourriez envisager l'emploi du pilote de périphérique ANSI.SYS pour ajouter la couleur. Avec une version DOS 3.3 ou ultérieure, vous pouvez également ajouter le paramètre /L à la commande BACKUP, créant ainsi un fichier journal (qui documente les opérations de sauvegarde). Un autre perfectionnement à considérer est l'addition d'un test des codes de sortie par IF ERRORLEVEL pour afficher des messages d'erreur en cas d'échec de l'opération de sauvegarde.

CHAPITRE 9

Emploi optimal d'AUTOEXEC.BAT

PERSONNALISER AUTOEXEC.BAT

A présent, vous avez vu toutes les commandes DOS de traitement par lots, appris à écrire des programmes à l'aide de DEBUG et créé plusieurs fichiers de commandes complexes. Regardons quelques fichiers de commandes amusants que vous pourriez éventuellement exécuter à partir de votre fichier AUTOEXEC.BAT. Chaque fichier de commandes de ce chapitre fait appel à un programme simple que vous pouvez créer avec DEBUG.

Un fichier de commande appelé BONJOUR.BAT utilise le programme JOURSEM.COM pour déterminer le jour de la semaine. Le fichier de commandes affiche ensuite un message approprié à chaque jour. Voici ce fichier de commandes :

```
ECHO OFF
REM Détermine le jour de la semaine à l'aide
REM du programme JOURSEM.COM. Ce programme
REM renvoie un code de sortie que vous testez
REM par la condition IF ERRORLEVEL. Dimanche
REM a la valeur 0, Lundi la valeur 1, etc.
REM Samedi a la valeur 6. Une fois le jour
REM défini, le fichier de commande affiche le
REM message approprié.
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 1 GOTO
                                   DIMANCHE
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO
                                   LUNDI
```

```
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 GOTO
                                  MARDI
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO
                                  MERCREDI
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 GOTO
                                  JEUDI
IF ERRORLEVEL 5 IF NOT ERRORLEVEL 6 GOTO
                                  VENDREDI
IF ERRORLEVEL 6 GOTO SAMEDI
:DIMANCHE
ECHO C'est Dimanche.
ECHO Allez donc faire du vélo.
GOTO FIN
:LUNDI
ECHO Déjà Lundi.
ECHO La semaine sera dure.
GOTO FIN
:MARDI
ECHO Nous sommes aujourd'hui Mardi.
ECHO La semaine commence à peine.
GOTO FIN
:MERCREDI
ECHO Voilà Mercredi.
ECHO Bravo ! la bosse est passée.
GOTO FIN
:JEUDI
ECHO Seulement Jeudi.
ECHO Encore un petit effort !
GOTO FIN
:VENDREDI
ECHO Enfin Vendredi.
ECHO Le bonheur !
GOTO FIN
:SAMEDI
ECHO Mais oui : Samedi !
ECHO Vous devriez être au lit.
:FIN
```

Vous pouvez noter des rendez-vous ou une réunion importante en modifiant simplement la commande ECHO appropriée (soit le jour-même, soit la veille). Le programme JOURSEM.COM utilise le service DOS *Get Date* (fonction

lecture de la date système) pour déterminer le jour de la semaine. Le programme retourne une valeur du code de sortie égale à 0 pour Dimanche, 1 pour Lundi, et ainsi de suite. Vous pouvez créer JOURSEM.COM avec DEBUG, de la façon suivante :

```
C> DEBUG JOURSEM.COM
Fichier introuvable

-A 100
2AF5:0100 MOV AH, 2A
2AF5:0102 INT 21
2AF5:0104 MOV AH, 4C
2AF5:0106 INT21
2AF5:0108
-R CX
CX 0000
:8
-W
Ecriture de 0008 octets
-Q

C>
```

De façon analogue, le fichier de commandes suivant, appelé CALEND.BAT, utilise le programme MOIS.COM pour déterminer le mois actuel. Le mois étant connu, le fichier de commandes effectue un branchement vers des commandes spécifiques de ce mois et affiche une liste de dates et de réunions :

```
ECHO OFF
REM Détermine le mois courant à l'aide du
REM programme MOIS.COM. Le programme renvoie
REM une valeur du code de sortie de 1 à 12
REM qui représente le mois courant (1 pour
REM janvier, 12 pour décembre). Une fois le
REM mois déterminé, branche vers le mois
REM spécifié et affiche les dates importantes
REM de ce mois.
MOIS
```

```
IF ERRORLEVEL  1 IF NOT ERRORLEVEL 2 GOTO
                                JANVIER
IF ERRORLEVEL  2 IF NOT ERRORLEVEL 3 GOTO
                                FEVRIER
IF ERRORLEVEL  3 IF NOT ERRORLEVEL 4 GOTO
                                MARS
IF ERRORLEVEL  4 IF NOT ERRORLEVEL 5 GOTO
                                AVRIL
IF ERRORLEVEL  5 IF NOT ERRORLEVEL 6 GOTO
                                MAI
IF ERRORLEVEL  6 IF NOT ERRORLEVEL 7 GOTO
                                JUIN
IF ERRORLEVEL  7 IF NOT ERRORLEVEL 8 GOTO
                                JUILLET
IF ERRORLEVEL  8 IF NOT ERRORLEVEL 9 GOTO
                                AOUT
IF ERRORLEVEL  9 IF NOT ERRORLEVEL 10 GOTO
                                SEPTEMBRE
IF ERRORLEVEL 10 IF NOT ERRORLEVEL 11 GOTO
                                OCTOBRE
IF ERRORLEVEL 11 IF NOT ERRORLEVEL 12 GOTO
                                NOVEMBRE
IF ERRORLEVEL 12 GOTO DECEMBRE

:JANVIER
ECHO 1er Janvier - Nouvel an
GOTO FIN

:FEVRIER
GOTO FIN

:MARS
GOTO FIN

:AVRIL
ECHO 15 Avril - Pâques
ECHO 16 Avril - Lundi de Pâques
GOTO FIN

:MAI
ECHO 1er Mai - Fête du travail
ECHO   8 Mai - Armistice 1945
ECHO  24 Mai - Ascencion
GOTO FIN
```

```
:JUIN
ECHO  3 Juin - Pentecôte
ECHO  4 Juin - Lundi de Pentecôte
GOTO FIN

:JUILLET
ECHO 14 Juillet - Fête nationale
GOTO FIN

:AOUT
ECHO 15 Aôut - Assomption
GOTO FIN

:SEPTEMBRE
GOTO FIN

:OCTOBRE
GOTO FIN

:NOVEMBRE
ECHO  1er Novembre - Toussaint
ECHO   11 Novembre - Armistice 1918
GOTO FIN

:DECEMBRE
ECHO   25 Décembre - Noël
GOTO FIN

:FIN
```

Dans cet exemple, le fichier de commandes contient les jours fériés, mais il est facile, avec la commande ECHO, d'ajouter les anniversaires et autres dates importantes. Pour créer le programme MOIS.COM avec DEBUG, procédez comme ceci :

```
C> DEBUG MOIS.COM
Fichier introuvable

-A 100
2AF5:0100 MOV AH, 2A
2AF5:0102 INT 21
2AF5:0104 MOV AL, DH
2AF5:0106 MOV AH, 4C
```

```
2AF5:0108 INT 21
2AF5:010A
-R CX
CX 0000
:A
-W
Ecriture de 000A octets
-Q

C>
```

Dans certains cas, votre traitement par lots sera spécifique d'un certain jour du mois, par exemple le 15. Le programme suivant JOURMOIS.COM, renvoie une valeur du code de sortie qui contient le jour du mois :

```
C> DEBUG JOURMOIS.COM
Fichier introuvable

-A 100
2AF5:0100 MOV AH, 2A
2AF5:0102 INT 21
2AF5:0104 MOV AL, DL
2AF5:0106 MOV AH, 4C
2AF5:0108 INT 21
2AF5:010A
-R CX
CX 0000
:A
-W
Ecriture de 000A octets
-Q

C>
```

Les deux commandes suivantes vous montrent comment l'on peut déterminer le 15 du mois après exécution du programme JOURMOIS.COM par le fichier de commandes :

```
JOURMOIS
IF ERRORLEVEL 15 IF NOT ERROR LEVEL 16 GOTO 15
```

Les utilisateurs choisissent souvent de détruire les copies périmées ou inutiles de leurs fichiers à des dates spécifiques.

A l'aide du programme JOURMOIS.COM, votre fichier AUTOEXEC.BAT déterminera si ce jour est le jour de destruction des fichiers et, si oui, il les détruit.

Le fichier de commandes HEURJOUR.BAT utilise le programme HEURE.COM pour déterminer l'heure courante et affiche un message approprié à cette heure :

```
ECHO OFF
REM Détermine l'heure courante à l'aide du
REM programme HEURE.COM, qui renvoie une
REM valeur du code de sortie entre 0 et 23
REM (0 pour minuit, 1 pour une heure etc.).
REM Une fois l'heure connue, affiche un
REM message spécifique.
HEURE
IF ERRORLEVEL  0 IF NOT ERRORLEVEL 6 GOTO TT
IF ERRORLEVEL  6 IF NOT ERRORLEVEL 12 GOTO M
IF ERRORLEVEL 12 IF NOT ERRORLEVEL 18 GOTO A
IF ERRORLEVEL 18  GOTO S
:TT
ECHO Vous êtes debout trop tôt ou trop tard.
GOTO FIN
:M
ECHO Bonjour.
GOTO FIN
:A
ECHO Bonne après-midi
GOTO FIN
:S
ECHO Bonsoir. Il commence à se faire tard.
:FIN
```

Le programme HEURE.COM utilise le service DOS *Get Time* (fonction lecture de l'heure) pour déterminer l'heure courante. Vous créez ce programme avec DEBUG :

```
C> DEBUG HEURE.COM
Fichier introuvable

-A 100
2AF5:0100 MOV AH, 2C
2AF5:0102 INT 21
2AF5:0104 MOV AL, CH
```

```
2AF5:0106 MOV AH, 4C
2AF5:0108 INT 21
2AF5:010A
-R CX
CX 0000
:A
-W
Ecriture de 000A octets
-Q

C>
```

De même, le programme MINUTE.COM renvoie les minutes de l'heure courante :

```
C> DEBUG MINUTE.COM
Fichier introuvable

-A 100
2AF5:0100 MOV AH, 2C
2AF5:0102 INT 21
2AF5:0104 MOV AL, CL
2AF5:0106 MOV AH, 4C
2AF5:0108 INT 21
2AF5:010A
-R CX
CX 0000
:A
-W
Ecriture de 000A octets
-Q

C>
```

Avec ces outils en main, vous pouvez libérer toute la puissance de vos fichiers de commandes. Modifiez vos fichiers lorsque vos besoins évoluent. Expérimentez les programmes présentés dans ce chapitre ; j'espère qu'ils vous plairont.

Annexes

ANNEXE A :
TABLEAU DES CODES DE SORTIE DOS

Plusieurs commandes MS-DOS délivrent des codes de sortie que vous pouvez tester par la commande IF ERRORLEVEL. Le tableau suivant donne les noms de ces commandes et les codes de sortie correspondants :

Nom	Code	Signification
BACKUP	0	Sauvegarde réussie
BACKUP	1	Aucun fichier à sauvegarder
BACKUP	2	Conflits de partage, sauvegarde incomplète
BACKUP	3	Arrêt Ctrl+C par l'utilisateur, sauvegarde incomplète
BACKUP	4	Erreur fatale, sauvegarde incomplète
DISKCOMP	0	Aucune différence entre les disquettes
DISKCOMP	1	Disquettes de types différents
DISKCOMP	2	Arrêt Ctrl+C par l'utilisateur, comparaison interrompue
DISKCOMP	3	Erreur irrémédiable en lecture ou écriture
DISKCOMP	4	Mémoire insuffisante, lecteur non valide ou erreur de syntaxe
DISKCOPY	0	Copie réussie
DISKCOPY	1	Erreur non fatale en lecture ou écriture
DISKCOPY	2	Arrêt Ctrl+C par l'utilisateur, opération diskcopy incomplète
DISKCOPY	3	Impossible de lire la disquette source ou de formater la disquette cible

Nom	Code	Signification
DISKCOPY	4	Mémoire insuffisante, lecteur non valide ou erreur de syntaxe
FORMAT	0	Formatage réussi
FORMAT	3	Arrêt Ctrl+C par l'utilisateur, formatage incomplet
FORMAT	4	Erreur fatale, formatage incomplet
FORMAT	5	Réponse Non de l'utilisateur à la question : «Poursuite du formatage (O/N) ?»
GRAFTABL	0	Commande réussie
GRAFTABL	1	Table précédente remplacée par la nouvelle
GRAFTABL	2	Aucune nouvelle table chargée
GRAFTABL	3	Paramètre incorrect sur ligne de commande
GRAFTABL	4	Version DOS incorrecte (4.0 requise)
KEYB	0	Chargement du clavier réussi
KEYB	1	Ligne de commande non valide
KEYB	2	Tableau de définition du clavier non valide
KEYB	3	Impossible de créer le tableau
KEYB	4	Erreur périphérique : console
KEYB	5	Page de codes non préparée
KEYB	6	Tableau de traduction manquant
KEYB	7	Version DOS incorrecte
REPLACE	0	Remplacement réussi
REPLACE	2	Fichier source non trouvé
REPLACE	3	Chemin source ou cible non trouvé
REPLACE	5	Fichier cible pour lecture seule
REPLACE	8	Mémoire insuffisante
REPLACE	11	Ligne de commande non valide
REPLACE	15	Spécification du lecteur non valide

Nom	Code	Signification
REPLACE	22 *	Version DOS incorrecte
RESTORE	0	Restauration des fichiers réussie
RESTORE	1	Aucun fichier à restaurer
RESTORE	2 *	Conflits de partage, restauration incomplète
RESTORE	3	Arrêt Ctrl+C par l'utilisateur, restauration incomplète
RESTORE	4	Erreur fatale, restauration incomplète
XCOPY	0 ♦	Copie sans erreur
XCOPY	1 ♦	Aucun fichier à copier
XCOPY	2 ♦	Arrêt Ctrl+C, XCOPY incomplète
XCOPY	4 ♦	Erreur d'initialisation (mémoire insuffisante, lecteur non valide, fichier ou chemin non trouvé ou erreur de syntaxe)
XCOPY	5 ♦	Une erreur INT 24 s'est produite

* MS-DOS ne renvoie pas ce code.

♦ PC-DOS ne renvoie pas ce code.

ANNEXE B : TABLEAU DES COMMANDES ANSI.SYS

Tout au long de ce livre, nous avons fait un usage extensif des séquences d'échappement ANSI.SYS. Le tableau ci-dessous résume les commandes ANSI.SYS :

Séquence	Fonction
Escape[*lignes* A	Déplace le curseur vers le haut, du nombre de lignes spécifié
Escape[*lignes* B	Déplace le curseur vers le bas, du nombre de lignes spécifié

Séquence	Fonction
Escape[*colonnes* C	Déplace le curseur vers la droite, du nombre de colonnes spécifié
Escape[*colonnes* D	Déplace le curseur vers la gauche, du nombre de colonnes spécifié
Escape[*ligne*;*col* H	Amène le curseur à la position ligne, colonne
Escape[*ligne*;*col* f	Comme ci-dessus
Escape[*ligne*;*col* R	Renvoie la ligne et la colonne actuelles (note : séquence non reconnue par MS-DOS)
Escape[s	Sauvegarde la position du curseur
Escape[u	Rétablit la position du curseur
Escape[2J	Efface l'écran. Ramène le curseur à la position origine.
Escape[K	Efface la fin de ligne à partir du curseur.
Escape[*couleur* m	Détermine l'attribut de couleur d'écran
Escape[6n	Renvoie la position du curseur
Escape[*mode* l *	Rétablit le mode vidéo initial
Escape[=*mode* h	Définit le mode vidéo
Escape[=7l *	Inhibe le retour automatique à la fin de la ligne
Escape[0;*touche*; "chaîne"p	Définit une touche de fonction
Escape[0q **	Ignore les commandes tentant d'affecter des touches supplémentaires d'un clavier étendu
Escape[1q ***	Permet d'affecter les touches supplémentaires d'un clavier étendu, même si ANSI.SYS a été chargé sans le commutateur /X

* Le dernier caractère est un *l* minuscule (ni le *L* majuscule, ni le chiffre un).

** Nouveau avec PC-DOS version 4.0 ; n'existe pas sur MS-DOS.

*** Notez que les caractères après [sont le chiffre un et le *q* minuscule.

Index

Caractères spéciaux

A

B

C

D

E

F

O

P

R

S

T

V

Achevé d'imprimer en décembre 1990
sur les presses de Cox and Wyman Ltd
(Angleterre)

Dépôt légal : janvier 1991
Imprimé en Angleterre

MICRO-INFORMATIQUE



QUICK REFERENCE

Le "Quick Reference" s'adresse à tout utilisateur d'un logiciel ou d'un système. Vous trouverez dans un format de poche toutes les instructions ou commandes nécessaires à leur mise en œuvre. Des commentaires pratiques vous donneront les avantages ou les pièges à éviter pour chaque procédure. Le caractère concis et complet du "Quick Reference" en fait un ouvrage destiné à vous fournir une assistance permanente à portée de la main.

MS-DOS, FICHIERS BATCH • KRIS JAMSA

Pour vous aider à mettre au point des fichiers *Batch* et à les maîtriser parfaitement, ce guide insiste sur les points suivants : créer vos premiers fichiers de commandes et les exécuter, programmer des fichiers *Batch* selon des techniques plus évoluées ; optimiser l'emploi de ANSI.SYS et de AUTOEXEC.BAT ; écrire des *Batch* plus performants avec DEBUG ; créer une interface utilisateur en *Batch* ; gérer l'imprimante et le disque dur, adapter les couleurs à l'écran et les touches de fonction ; exploiter votre bibliothèque de fichiers *Batch*.

Il existe trois autres collections Micro-Informatique :

INITIATION FACILE
Collection de prise en main qui constitue l'entrée dans la gamme.

GUIDE RAPIDE
Destinée à l'apprentissage, elle propose un approfondissement des concepts vus par Initiation Facile.

MICRO MEMENTO
Destinée à un public voulant avoir un ouvrage de référence exhaustif sur un thème, elle joue donc le rôle de complément des deux autres collections.

68517.2
ISBN 2-266-04059-6



MICRO INFORMATIQUE

4533

MICROSOFT PRESS

QUICK REFERENCE

MS-DOS • FICHIERS BATCH

6

PRESSES POCKET